jamk.fi

Tietoverkon automatisointi

Ville Korhonen

Opinnäytetyö
Helmikuu 2016
Tietotekniikan koulutusohjelma
Tekniikan ja liikenteen ala
Tietoverkkotekniikka

Jyväskylän ammattikorkeakoulu
JAMK University of Applied Sciences

jamk.fi

Kuvailulehti

Tekijä(t)	Julkaisun laji	Päivämäärä
Korhonen, Ville	Opinnäytetyö, AMK	2/2016
	Sivumäärä 73	Julkaisun kieli Suomi
		Verkkojulkaisulupa myönnetty: X

Työn nimi
Tietoverkon automatisointi

Tutkinto-ohjelma
Tietotekniikka

Työn ohjaaja(t)
Karo Saharainen, Mika Rantonen

Toimeksiantaja(t)
Puolustusvoimat

Tiivistelmä

Opinnäytetyön tavoitteena oli tarkastella, kuinka verkkolaitteiden toimintoja voidaan suorittaa tietoturvallisesti ja automatisoidusti. Tarkoituksena oli ottaa huomioon KATAKRI II:n vaatimukset tietoturvalliselle tiedon käsittelylle ja sen siirtämiselle.

Tietoverkon laitteiden automatisointia tarkasteltiin TCL-ohjelmointikielen avulla. Jolla verkon laitteissa voidaan suorittaa automatisoituja toimintoja skriptien avulla. EEM-sovelluksilla laitteesta kerättiin tietoja, jotka raportoitiin eteenpäin keskitettyyn paikkaan.

Teoriaosuudessa keskityttiin järjestelmän automatisoinnin periaatteisiin, kuinka automatisointi tulisi toteuttaa yrityksen tietoverkossa. Teoriaosuudessa huomioitiin tiedonkäsittelyyn liittyvä tietoturvallisuus käyttäen lähteenä KATAKRI-dokumentaatiota.

Työn tuloksena saatiin esimerkkejä siitä, kuinka yrityksen lähiverkossa voidaan tietoturvallisesti toteuttaa verkon automatisointia. Ja kuinka verkkolaitteisiin voidaan luoda herätteisiin reagoivia sovelluksia, jotka keräävät laitteesta tietoja ja välittävät sitä eteenpäin. Opinnäytetyötä voidaan soveltaa ympäristöissä, joissa on tarvetta kerätä verkkolaitteista tilatietoja automatisoidusti.

Avainsanat (asiasanat)
Automatisointi, Skriptaus, Tietoturva, TCL, EEM

Muut tiedot

jamk.fi

Description

Author(s)	Type of publication	Date
Korhonen, Ville	Bachelor's thesis	2/2016
		Language of publication: Finnish
	Number of pages: 73	Permission for web publication: X

Title of publication
Network automation

Degree programme
Information Technology

Supervisor(s)
Karo Saharinen, Mika Rantonen

Assigned by
Finnish Defence Forces

Abstract

The aim of this thesis was to examine how network actions can be run safely and automatically. The purpose was to examine KATAKRI II concerning the safety of transfer and information processing in network. Network automation was examined through TCL programming language which is able to, take automated actions through scripts. The information was gathered and reported with EEM applications in one centralized location.

The focus of the theoretical part is on the principles of the automation system and the implementation methods of automation in the network. Data processing and data security were reviewed in the theoretical part using KATAKRI documentation as a source.

The results present an example on how to securely implement network automation in the local network of the company and how to create applications that are responsive gather information from network devices and forward it. This thesis can be applied in the environments where there is need to collect status information in automated way from the network devices.

Keywords/tags (subjects)
Automation, Scripting, Network security, TCL, EEM

Miscellaneous

Sisältö

Kuviot

Lyhenteet

AAA	Authentication Authorization Accounting
API	Application Programming Interface
BIOS	Basic Input-Output System
EEM	Embedded Event Manager
EMM	Embedded Menu Manager
ESM	Embedded Syslog Manager
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
ICMP	Internet Control Message Protocol
iLO	Integrated Lights Out
IP	Internet Protocol
IVR	Interactive Voice Response
KATAKRI	Kansallinen turvallisuusauditointikriteeristö
RADIUS	Remote Authentication Dial In User Service
SCP	Secure Copy
SDN	Software-Defined Networking
SSH	Secure Shell
TACACS	Terminal Access Controller Access Control System
TCL	Tool Command Language
USGCB	The United States Government Configuration Baseline

1 Työn lähtökohdat

1.1 Toimeksiantaja

Opinnäytetyön toimeksiantajana toimi Suomen puolustusvoimien pääesikunnan alainen joukko-osasto, jonka laitoksen pääosat on sijoitettu Helsinkiin ja Jyväskylään. Puolustusvoimien tehtävänä on Suomen sotilaallinen puolustaminen, muiden viranomaisten tukeminen ja osallistuminen kansainväliseen kriisinhallintaan. Sotilaallisen puolustukseen kuuluu maa-alueiden, ilmatilan, vesialueiden ja alueellisen koskemattomuuden turvaaminen, kansalaisten perusoikeuksien ja elinmahdollisuuksien turvaaminen sekä sotilaskoulutuksen antaminen. Muiden viranomaisten tukemisella Puolustusvoimat antaa virka-apua turvallisuuden ja yleisen järjestyksen ylläpitämiseksi sekä estämään terrorismirikosta. (Puolustusvoimien tehtävät 2013).

1.2 Tavoitteet

Opinnäytetyön tavoitteena oli tutkia tietoverkon aktiivilaitteiden toimintojen automatisointia tietoturvallisesti. Toimintojen automatisointi tapahtui monitoroimalla verkkolaitteita sekä keräämällä niistä tilatietoja keskitettyyn paikkaan ja tekemään niiden avulla erilaisia päätöksiä. Tilatietojen kerääminen ja toimintojen toteuttaminen tuli suorittaa tietoturvallisesti ottaen huomioon Katakri II-vaatimus. Työssä tutkittiin, mitä tietoa verkkolaitteista voitiin kerätä automaattisesti, ja kuinka niiden avulla saatiin automatisoida verkkolaitteita tietoturvallisesti.

Työssä keskityttiin kehitystyyppiseen tutkimustyöhön, jossa keskeisenä osana on käytännön toteutus ja teoriaosuus. Käytännön toteutuksessa tarkoituksena oli antaa viitemalleja ja menetelmiä, kuinka tietoverkkoa voidaan automatisoida tietoturvallisesti. Teoriaosuudessa keskityttiin aiheen kannalta olennaisiin aihealueisiin, jotka liittyvät tietoverkkojen automatisointiin. Haasteeksi työssä muodostui aiheen rajaaminen. Aiheeseen liittyi monta automatisoinnin kannalta käytettävää menetelmää ja ominaisuutta. Aihetta rajattiin käytännössä ja teoriaosuudessa niin, että pääpaino tutkimuksena oli, kuinka tieto voitiin siirtää laitteiden välillä tietoturvallisesti. Työssä

tutkittiin tietoverkon aktiivilaitteiden ohjelmallista automatisointia ns. skriptaamista. Käytetty ohjelmointikieli työssä oli TCL. Ohjelmointikieli valittiin, koska Cisco-verkonaktiivilaitteet tukevat sitä, eli laitteissa pystyttiin paikallisesti suorittamaan TCL-kielellä tehtyjä komentoja, ja koska opinnäytetyön testausympäristö koostui Ciscon verkkolaitteista.

1.3 Tietoperusta

Työssä lähteinä käytettiin laitevalmistajien dokumentaatioita ns. *white paper*-dokumentteja, IT-alan tieteellisiä julkaisuja sekä oman osaamisen lisäksi aiheeseen liittyvää kirjallisuutta. Laitevalmistajien dokumenteissa ohjeistetaan, kuinka laiteominaisuuksia otetaan hallitusti käyttöön. IT-alan tieteellisistä julkaisuista haettiin teoreettista ja tieteellistä perustaa laiteominaisuuksille. Kirjallisilla julkaisuilla perehdyttiin materiaaleihin, joita ei löytynyt e-kirjoina tai muuna elektronisessa muodossa olevana kirjallisuutena. Lähteitä lähestyttiin kriittisesti ottaen huomioon niiden ajantasaisuus, lähteen alkuperä sekä materiaalin tieteellinen tausta ja paikkansapitävyys.

2 TIETOVERKON AUTOMATISOINTI

2.1 Yleistä

Tietoverkoista on tullut kriittinen osa nykypäivän yritysten ja ihmisten elämää. Koska tietoverkko ei ole ihmisille näkyvä osa päivittäistä toimintaa, on huomio keskittynyt enemmän saataviin palveluihin kuten sähköposti, sosiaalinen media ja reaaliaikaiset suoratoistopalvelut. Tietoverkon roolia vähätellään tai unohdetaan jopa kokonaan. Kuitenkin tietoverkolla on merkittävä rooli yritysten tehokkuuteen ja yleiseen toimintaan. (Jones 2005, 1).

Tietoverkon hallitseminen, ylläpito ja laitevioista toipuminen suoritetaan useissa yrityksissä käsin eli manuaalisesti. Jopa yksinkertaisimmat toiminnot, kuten laiteasetusten varmuuskopiointi saatetaan suorittaa käsin. Jotta tietoverkko voidaan pitää toiminnassa näissä tapauksissa, pitää kaikki toiminnot suorittaa manuaalisesti ja joskus

jopa paikan päältä. Tietoverkon automatisointi eli päivittäisten toimintojen kuten varmuuskopiointi ja muutosten hallinta, voidaan suorittaa automaattisesti ilman, että siihen tarvitsee ihmisen puuttua. Pitkälle automatisoidussa ympäristössä voidaan palautua laiteviasta ilman, että se vaikuttaa yrityksen palveluihin. (Jones 2005, 1).

Elisalla kuitukaapelin katkeaminen Lohjalla aiheutti valtakunnallisesti ongelmia Elisan ja Saunalahden asiakkaiden yritysverkkojen liittymissä, kiinteissä ja matkapuhelinverkoissa sekä kansainvälisissä puhepalveluissa. Kuitukaapelin katkeamisen takia yhteydet Elisan verkon ja muun maailman välillä olivat toimimattomina useita tunteja. Elisan automaattisen uudelleenohjauksen toimimattomuus aiheutti suuren katkon Elisan verkkoon. Elisan palvelunhallinnan johtaja Sami Komulaisen mukaan "Teimme muutoksia Tukholman laitetilassa ja korvasimme joitain laitteita". Tästä voidaan päätellä, että Elisa on toteuttanut puutteellisesti muutosten hallinnan oman runkoverkon ja Tukholman välisissä laitteissa. (Juntunen 2014; Hartig 2014).

Ristiriitaisella toimintatavalla toteutettu manuaalinen verkonhallinta voi pahimmillaan johtaa tietoturvariskeihin ja verkon hallittavuuden menettämiseen. Tällaisesta hyvä esimerkki on, kun samanlaisella pohjalla toteutetaan kaksi verkkolaitetta. Toisella laitteella toteutus toimii odotetulla tavalla, toisella laitteella havaitaan ongelmia, esimerkiksi verkkoliikenteen katkeaminen ja hallintayhteyden menettäminen. Tällainen ongelma voi esiintyä kahden eri laitevalmistajan välillä sekä saman laitevalmistajan eri ohjelmistoversioiden kesken.

Ristiriitaisessa verkonhallinnassa ja manuaalisessa ylläpidossa inhimillisen virheen riski on suuri, etenkin jos kyseessä on laaja verkkoympäristö. Verkkolaite voi käyttöönottovaiheessa toimia normaalisti ja kaikki toiminnot voivat vaikuttaa normaalilta. Suuressa ympäristössä virheellisesti hoidetut asetukset voivat tulla esille vasta myöhemmin. Ristiriitainen verkonhallinta aiheuttaa turvallisuusriskejä, hallittavuuden ja skaalautuvuuden alenemista sekä ongelmatilanteista palautumista. Turvallisuusriskeihin voivat johtaa esimerkiksi laajassa ympäristössä huonosti dokumentoitu ja toteutetut palomuuriasetukset. Laajassa ympäristössä käsin toteutetut asetukset useaan laitteeseen voi johtaa väärin asetettuun palomuurisääntöön. Täten palomuuriin voi jäädä esimerkiksi väärä portti auki, ja tämä voi johtaa tietoturvariskeihin. Ris-

tiriitaisesti toteutettu verkonhallinta ja skaalautuvuus voi olla hankala ylläpitää. Suuressa verkkoympäristössä monessa laitteessa voi olla erilaiset asetukset. Pienikin muutos asetuksissa voi aiheuttaa suuria riskejä verkon toiminnollisuuteen. Ongelmatilanteista palautumista hankaloittaa, kun laitteille ei ole johdonmukaisesti tehty käyttöönottosuunnitelmaa. (Jones 2005, 4).

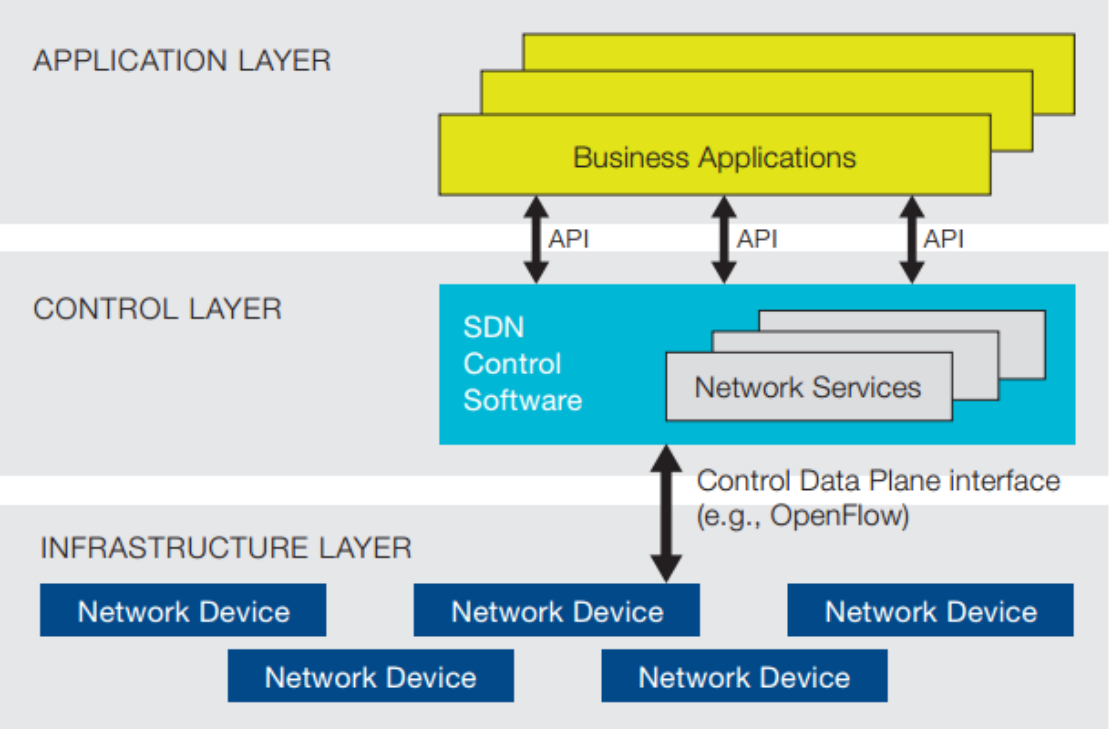
2.2 SDN

SDN arkkitehtuurissa data- ja hallintatasot on eroteltu toisistaan. Erottelun etuna verkon hallinta voidaan loogisesti keskittää yhteen paikkaan. Tuloksena verkkoa voidaan hallinnoida ja automatisoida tehokkaammin vastaamaan nykypäivän tarpeita jatkuvasti skaalautuvasta tiedonsiirtoverkosta. (ONF 2012).

ONF (Open Networking Foundation) on voittoa tavoittelematon yhteisö, joka kehittää SDN-arkkitehtuuria ja standardoi sen kriittisiä elementtejä, kuten OpenFlow-protokollaa. OpenFlow-protokollan avulla luodaan keino kommunikoida data-ja hallintatasojen välillä tietoverkonlaitteissa. (ONF 2012).

Perinteisessä tietoverkon arkkitehtuurissa verkkorakenne toteutetaan hierarkkisesti. Verkkorakenne muodostuu puumallisesta verkkotopologiasta, jossa ethernet-kytkimiä on kerroksittain. Tämä arkkitehtuuri sopi, kun tiedonsiirto käytiin työasema-palvelintyyppisessä tiedonsiirrossa. Nykypäivän yrityksen tiedonsiirrossa työasemat muodostavat useita yhteyksiä tietokantoihin ja palvelimiin. Tämä luo haasteita nykypäivän tietoverkon arkkitehtuurille, kun tietoa voidaan kerätä ja siirtää usean palvelimen välillä luoden useita tietoliikenneyhteyksiä, ennen kuin käyttäjä pääsee tietoon käsiksi. Tämän vuoksi yritykset harkitsevat tiedon käsittelyyn ja sen säilytykseen pilvipohjaista ratkaisua, jolla voidaan vähentää yhden käyttäjän luomaa tietoliikennemäärää. Tietoverkkojen ylläpitäjät kohtaavat nykypäivänä suurimpana haasteena täyttää tiedonsiirtoverkolle asetetut vaatimukset, kun yrityksissä järjestelmien ylläpitoon ja kehitykseen tarkoitettua budjettia leikataan jatkuvasti. Ylläpitäjät joutuvat käyttämään laitetason hallintaa ja manuaalisia prosesseja verkon ylläpitämiseen. (ONF 2012).

SDN-arkkitehtuurissa hallintataso eriyttämällä, mahdollistetaan arkkitehtuurin alempien kerroksien skaalautuvuus ja muokattavuus sovelluksille ja tietoverkon palveluille (ks. Kuvio 1). (ONF 2012)



Kuvio 1. SDN-arkkitehtuuri (ONF 2012)

SDN-arkkitehtuurissa tietoverkon looginen hallinta on keskitetty SDN-hallintasovellukselle. Hallintasovellus ylläpitää loogista kuvaa koko tiedonsiirtoverkosta. Tuloksena sovellukset ja ohjelmat näkevät tietoverkon yhtenä suurena kytkimenä. SDN:n avulla yritykset ja operaattorit saavat valmistajasta riippumattoman hallinnan koko tiedonsiirtoverkolle yhdestä loogisesta pisteestä, mikä suuresti helpottaa verkon suunnittelua ja operointia. Tärkeimpänä ominaisuutena verkkoa voidaan hallinnoida ohjelmallisesti, jolloin ei tarvitse ylläpitää tuhansia riviä konfiguaatiota. Ohjelmallisesti hallittavassa verkossa muutokset voidaan tehdä ja ottaa käyttöön nopeasti. SDN-arkkitehtuurissa ei tarvitse tehdä usealle laitteelle erikseen muutoksia asetuksiin, muutokset tehdään yhteen loogiseen paikkaan. (ONF 2012).

SDN-arkkitehtuuri tukee API-rajapintaa. API-rajapinnan avulla voidaan implementoida yleisiä tietoverkon palveluita, kuten reitittämistä, multicast-tiedonsiirtoa, tietoturvakomponentteja, kaistan hallintaa, liikenteen muokkaamista, QoS, optimoida prosessointia ja tallennustilan käyttöä. SDN-arkkitehtuuria voidaan soveltaa niin langalliseen kuin myös langattomaan tiedonsiirtoon. (ONF 2012).

2.3 Automatisoinnin hyödyt

Manuaalisesti toteutetussa hallinnassa jokaiselle verkonlaitteelle muutokset tehdään per-laite periaatteella, eli jokaiselle yksittäiselle laitteelle tehdään muutokset tarpeen mukaan. Muutokset toteutetaan yleensä näin, koska laitteet on tehty niin hallittaviksi. Esimerkiksi reitittimille voidaan tehdä *black hole*-reittejä, jolloin ei-haluttu liikenne tiputetaan pois. Cisco IOS laitteissa reitit ohjataan *null0*-rajapintaan. Näin jokaiselle ei halutulle verkolle jouduttaisin luomaan oma reitti, esim.

ip route 100.0.0.0 255.0.0.0 null0

Manuaalisesti toteutettuna tämä tehtäisiin käsin jokaiselle Cisco IOS-reitittimelle. Kun ei-haluttuja reittejä on paljon, konfiguraation määrä kasvaa useita satoja rivejä suureksi. Kun rivejä on paljon, myös virheen todennäköisyys on suuri. Automatisoidussa verkossa muutosten hallinnan avulla voidaan tälle toimenpiteelle luoda oma politiikka. Politiikalle voidaan tehdä oma työjono, joka tarkastaa politiikassa olevat asetukset. Työjonon avulla verkosta skannataan kaikki laitteet, joihin politiikka voidaan asettaa. Työjonolla tarkastetaan, onko kaikilla laitteilla politiikkaan täsmäävät asetukset. Jos ei ole, laitteelle asetetaan politiikan mukaiset laiteasetukset. Näin asetukset tarvitsee tehdä vain kerran, ja työjono asettaa jokaiselle verkon laitteelle asetukset automaattisesti yhdenmukaiseksi. (Jones 2005, 20–21).

Skriptaaminen

Komentosarjat eli skriptaaminen on yksi tietoverkon automatisoinnin osa-alue. Skriptaamalla voidaan hallitusti muokata useita tietoverkon laitteita samanaikaisesti tekemällä jollain komentosarjakielellä eli skriptikielellä komentoja. Komennoilla voidaan laitteille asettaa tai hakea tietoja laitteelta. Skriptaustavat ovat yleensä kahta eri päätyyppiä: verkonlaitteella tapahtuva skriptaaminen ja palvelimella tapahtuva skriptaaminen. Erona näissä on se missä skriptikoodi tulkitaan. Verkonlaitteella tapahtuvassa skriptauksessa skriptit ovat yleensä tapahtumaohjattuja, ts. ne toteutetaan vasta jonkin tapahtuman toteutuessa paikallisella laitteella. Tämmöiset tapahtumat voivat olla esim. muutokset reititystaulussa, tai jokin ajastettu toiminto. Palvelinskriptauksessa palvelin voi saada usealta verkonlaitteelta esim. SNMP trap-viestin, jolloin palvelin viestin saatuaan tekee tarvittavat toiminnot usealle eri verkonlaitteelle. Tämmöisiä toimintoja voi olla viestin raportointi eteenpäin sähköpostilla, tai skriptin syöttäminen verkonlaitteille. (Lappalainen 2010; Jones 2005).

Muunneltavuus, Toiminnoille proseduurit

Automatisoinnin etuna on nopea ja luotettava vastaavuus verkon muutoksiin. Verkkoon voidaan tehdä nopeasti ja hallitusti muutoksia tietoturvallisesti, kun muutokset tehdään prosessinomaisesti. Prosessinomaisessa muutoksen hallinnassa muutoksia voidaan testata, katselmoida, hyväksyttää ja ajastaa käyttöön. Kun verkkomuutoksille on luotu tietty proseduuri, vähentää se virheellisten muutosten riskiä. Jokaisen prosessin vaiheen voi hyväksyttää eri henkilöllä. Prosessinomaisella tietoverkonhallinnalla voidaan seurata verkkoon tehtyjä muutoksia, josta on hyötyä jälkeenpäin vianselvityksessä. (Jones 2005, 14).

Prosessinomaisella muutoksen hallinnalla on myös ristiriitansa. Esimerkiksi, kun samalle verkonlaitteelle kaksi tai useampi eri henkilö tekee muutoksia voi syntyä laitteen asetuksiin ristiriita. Henkilöt tekevät muutokset verkonlaitteen alkuperäisten asetusten pohjalta, yksi muutoksista hyväksytään ennen muita ja otetaan tuotantoon. Muiden henkilöiden tekemät muutokset ovat ristiriidassa ensimmäiseksi hyväksyttyjen muutoksien kanssa. Ratkaisuna tähän on prosessiin luodut vaiheet ja ajas-

tettu toteutus. Kun muutokset lähetetään tarkasteltavaksi, muutosten tarkastajan tulee tiedottaa muita henkilöitä jotka tekevät muutoksia verkonlaitteille. *Näin muut henkilöt ovat tietoisia muista muutoksista.* Kun prosessin suoritetaan ajastetusti, vähennetään ristiriidan todennäköisyyttä. (Jones 2005, 15).

Johdonmukainen toteutus

Johdonmukaisella prosessilla muutokset suunnitellaan, hyväksytään, ja toteutetaan vain kerran. Prosessin etuna on että kaikki tehdään manuaalisesti vain yhden kerran. Jokaiselle laitteelle ei tarvitse erikseen tehdä samoja vaiheita. Johdonmukainen toimita parantaa verkon tietoturvallisuutta, hallittavuutta, sekä sen luotettavuutta. (Jones 2005, 15).

Palautuminen

Muutostenhallinta auttaa palautumaan ongelmatilanteista. Ongelmatilanteita voi syntyä äkillisestä laiteviasta, sähkökatkosta, tai väärin toteutetusta laitekonfiguraatiosta. Muutosten hallinnassa jokaisesta toimivasta laitekonfiguraatiosta tallennetaan varmuuskopio. Varmuuskopiota voidaan jälkeenpäin hyödyntää ongelmatilanteista toipumiseen. Muutosten hallinnassa varmuuskopiointi tulee suorittaa ajastetusti ja myös aina kun laitteelle tapahtuu muutoksia. Tällainen varmuuskopiointi voi olla käyttäjän tekemien muutoksien lisäksi lokitietojen monitorointi, autentikointi liikenne, tai SNMP-viestit. Varmuuskopiointia voidaan tehdä myös silloin kun laitteelle oletetaan tulevan muutoksia, esim. kun käyttäjä kirjautuu laitteelle sisään. (Jones 2005, 15-16).

Tehokkuus

Automatisoinnilla voidaan tehokkaasti muuttaa asetuksia useaan tietoverkonlaitteeseen samanaikaisesti. Manuaalisesti toteutettu muutos useaan laitteeseen on aikaa vievää ja vaivalloista. Jos 100 laitteeseen tulisi muuttaa laitteen salasana, tulisi ensiksi laskea kuinka paljon menee aikaa muuttaa yhden laitteen salasana, sen jälkeen

kertoa se laitteiden määrällä. Tämmöiseen toimenpiteeseen voi mennä hyvinkin kauan. Automatisoidussa verkossa muutostenhallinnan avulla toimenpide tarvitsee suorittaa vain kerran. Muutostenhallinnan avulla voidaan määrittää laitteelle tehtävät toimenpiteet, sekä mille laitteille toimenpide suoritetaan. (Jones 2005, 19).

Tiedon kerääminen

Tietoverkkoa tulee voida tarkastella ja kerätä tietoa verkonlaitteiden eri asetuksista. Automatisoidun tietoverkon etuna on, että sitä voidaan helposti ja jatkuvasti tarkastella. Manuaalisessa tietoverkonhallinnassa jokaisesta laitteesta täytyy erikseen hakea laitteen asetukset ja tarkastella niitä. Manuaalisesti toteutettuna tämä on virhealtista, kun tarkasteltavia laitteita on useita satoja, virheen todennäköisyys kasvaa. Jotkin laitteet voivat unohtua ja kaikkia asetuksia ei muista tarkastella, kun laitteen asetuksista voi tulla useita satoja rivejä tekstiä. Automatisoinnin avulla laitteista voidaan tarkastella vain sitä osaa asetuksista joihin on tarvetta. (Jones 2005, 19).

3 TIETOTURVA

Tässä luvussa perehdytään Katakrin teknisen tietoturvallisuuden osa-alueeseen ja sen vaatimuksiin. Opinnäytetyössä teknisen tietoturvallisuuden osa-alueesta otetaan huomioon osia, joissa käsitellään teknisesti kriittisen tiedon siirtämistä ja käsittelemistä. Työssä pyritään ottamaan huomioon tiedon saatavuus ja eheys, joilla varmistetaan tietoturvallinen tiedonsiirto ja käsittely järjestelmien välillä.

3.1.1 Yleistä

Katakri on auditointityökalu, joka on tarkoitettu viranomaisten käyttöön. Katakria voidaan käyttää kohdeorganisaation arviointiin sen kyvystä suojata salassa pidettä-

vää tietoa. Katakri on koonnos kansallisista säännöksistä ja kansainvälisiin perustuvista vähimmäisvaatimuksista. Katakriin on koottu vaatimuksia perustuvaan voimassa olevaan lainsäädäntöön ja kansainvälisesti Suomea sitoviin turvallisuusvelvoitteisiin. Katakri itse ei aseta vaatimuksia tietoturvallisuudelle. Keskeisin vaatimuslähde on asetus valtioneuvoston tietoturvallisuudesta valtionhallinnossa (681/2010), jäljempänä tietoturvallisuusasetus. Suomessa tietoturvallisuusasetusta noudatetaan niin kansallisen kuin myös kansainvälisen salassa pidettävän tiedon suojaamisessa. Katakrissa on käytetty EU:n neuvoston turvallisuussääntöjä (2013/488/EU), EU:n vähimmäisvaatimuksia ja perusperiaatteita turvallisuusluokitellun tiedon suojaamiseen. Katakrissa on lähdeviittaus jokaisen esitetyn vaatimuksen yhteydessä. (KATAKRI 2015b).

Katakri koostuu kolmesta eri osa-alueesta:

1. Turvallisuusjohtamisen osa-alueessa keskitytään organisaation turvallisuusjohtamisen valmiuksiin ja kykyyn.

2. Fyysisen turvallisuuden osa-alueessa kuvataan fyysistä käyttöympäristöä koskevia turvallisuusvaatimuksia salassa pidettävälle tiedolle. Fyysinen käyttöympäristö voidaan jakaa kolmeen alueeseen: tekninen turva-alue, turva-alue ja hallinnollinen alue.

3. Teknisen tietoturvallisuuden osa-alueessa kuvataan turvallisuusvaatimuksia tietojenkäsittely-ympäristölle. Tekninen tietoturvallisuus jaetaan käsiteltävän tiedon mukaiseen kolmeen alueeseen suojaustason mukaisesti.

- ST IV – Käyttö rajoitettu
- ST III – Luottamuksellinen
- ST II – Salainen
- ST I – Erittäin salainen

Vaatimukset mahdollistavat erilaisia toteutustapoja. Vaatimuksissa on esimerkkejä, kuinka vaatimuksen voi täydentää. Niissä kuvataan myös suosituksia ja parhaita käytäntöjä. Vaatimukset eivät ole sitovia. Suosituksia ja parhaita käytäntöjä löytyy VAHTI-ohjeista ja EU:n turvallisuussääntöjä koskevista ohjeasiakirjoista. (KATAKRI 2015b).

Katakria voidaan käyttää auditointityökaluna yrityksien, yhteisöjen ja viranomaisten turvallisuustyössä ja sen kehittämisessä. Katakrin avulla varmistetaan, että kohdeorganisaatiolla on salassa pidettävän tiedon oikeudettoman paljastumisen ehkäisemiseksi vaadittava ympäristö, jossa salassa pidettävää tietoa käsitellään. Katakrin avulla arvioidaan kohdeorganisaation kyky suojata salassa pidettävää tietoa. (KATAKRI 2015b).

3.1.2 Tekninen tietoturvallisuus

Katakrissa on kuvattu teknisen tietoturvallisuuden osa-alueeseen vaatimukset, joilla pyritään varmistamaan salassa pidettävän tiedon turvallisuusjärjestelyjen riittävyys sähköisissä käyttöympäristöissä. Teknisen tietoturvallisuuden vaatimukset on jaettu käyttö-, tietoaineisto-, tietojärjestelmä- ja tietoliikenneturvallisuuden alueisiin. Teknisen tietoturvallisuuden tiettyihin asiakokonaisuuksiin on lajiteltu niihin liittyviä vaatimuksia. Asiakokonaisuuksia ovat esimerkiksi hallintayhteydet, langattomat verkot, varmuuskopiointi ja etäkäyttö. (KATAKRI 2015a, 29).

Teknisen tietoturvallisuuden osa-alueeseen on myös kuvattu toteutusesimerkkejä, joilla voidaan saavuttaa useimmissa ympäristöissä hyväksyttävän suojauksen vähimmäistaso. Toteutusesimerkit voi korvata myös muulla vastaavan tason suojauksella. Toteutusesimerkeissä ei ole kuvattuna kaikkiin ympäristöihin tai muihin erikoitapauksiin riittäviä suojauksia. (KATAKRI 2015a, 29).

Kohdeorganisaation pitää pystyä todistamaan tilanteissa joissa suojauksille käytetään korvaavaa menettelyä, että menettelyllä saavutetaan riittävä suojaustaso. Jos kohdeorganisaation tavoitteena on saada toimivaltaisen virnaomaisen hyväksyntä tietojärjestelmälle, tulee toteutettujen suojausten olla riittäviä sekä toimivaltaisen viranomaisen että oman organisaation riskienarvioinnin havaintoihin nähden. (KATAKRI 2015a, 29).

Katakri suosittelee eriyttämään ja rajaamaan eri suojaustason käsittely-ympäristöt toisistaan mahdollisemman suppeaksi kustannusten hallitsemiseksi. Erityisesti sa-

lassa pidettävän tiedon ja viranomaisen salassa pidettävän tiedon käsittely-ympäristöt tulisi luokitella tarkoituksenmukaisesti. Esimerkiksi eriyttämällä suojaustason III ja suojaustason IV tiedon käsittely-ympäristöt, suojaustason III suojausmenetelmiä ja tekniikoita ei edellytetä toteutettavaksi suojaustason IV käsittely-ympäristössä. Tietojärjestelmien tarkastuksessa viranomaisena toimii ensisijaisissa tapauksissa Viestintävirasto. (KATAKRI 2015a, 29).

Tilanteessa, jossa arvioidaan vain sähköisessä muodossa tietoa käsittelevää tietojärjestelmää viranomaisen salassa pidettävän tiedon käsittely-ympäristössä, arvioidaan tällöin vain sähköiseen tietojenkäsittelyyn liittyvien vaatimusten täyttämistä. Esimerkiksi tietojenkäsittely joka koskee paperimuotoisen tiedon vaatimuksia, huomioidaan se vain soveltuvin osin. (KATAKRI 2015a, 29).

3.2 Järjestelmäturvallisuus

Tietoturvallisessa tiedonsiirrossa huomioitiin Katakrin teknisen tietoturvallisuuden asettamia vaatimuksia tietoturvalliselle tiedonsiirrolle ja tietoliikenneturvallisuudelle.

3.2.1 Hallintayhteydet

Tietoverkkolaitteisiin ja liittymiin tulisi olla hallintaoikeudet vain järjestelmän ylläpitäjillä tai vastaavilla. Tyypillisiä tietoverkkolaitteita ovat esimerkiksi kytkimet, reitittimet, palomuurit, langattomat tukiasemat, palvelimet, työasemat, Blade-runkojen-, ja ILO-hallintaliittymät. Hallintayhteydet ja hallintayhteyksiin käytettävät päätelaitteet tulisi rajata samalla suojaustasolle kuin käytettävä tietojenkäsittely-ympäristö. Yleisimmät hallintayhteystavat mahdollistavat pääsyn salassa pidettävään materiaaliin suoraan tai epäsuoraan. Esimerkiksi suorassa hallintayhteydessä tietokantojen ylläpitäjä pääsee suoraan käsiksi tietokantojen sisältöön. Epäsuorassa hallintayhteydessä ylläpitäjä pääsee muuttamaan tietojärjestelmää suojaavia palomuurisääntöjä. (KATAKRI 2015a, 35–36).

Matalamman suojaustason ympäristön hallinta voidaan toteuttaa ylemmän suojaustason hallinnalla. Kun matalamman suojaustason ympäristöä hallitaan ylemmän suojaustason ympäristöstä, tulee ottaa huomioon että ylemmän tason ympäristöstä ei pääse salassa pidettävää tietoa kulkemaan alemman tason ympäristöön. Suojaustasojen rajalla tulee olla viranomaisten kyseiselle suojaustasolle hyväksymä yhdyskäytävä ratkaisu. Ylemmän suojaustason ympäristön hallintayhteys alemman tason suojausympäristölle ei yleensä ole mahdollista, johtuen ylemmän tason hallintaliikenteen turvallisuuskriittisestä luonteesta. (KATAKRI 2015a, 35–36). (Viestintävirasto 2015a).

Vaatimuksina on että hallintayhteydet on rajattu suojaustasoittain. Hallintaliikenteen liikkuessa matalamman suojaustasonympäristössä sisältäen salassa pidettävää tietoa, tulee hallintaliikenteen olla salattu viranomaisten hyväksymällä salaustuotteella. Ylemmän tason hallintaliikenteen kulkiessa alemmalle tasolle, voidaan ko. suojaustason hallintatyöasema kytkeä hallittavaan laitteeseen fyysisesti käyttäen esim. konsolikaapelia, liikennekanavan kulkiessa luotettavasti fyysisesti suojatussa ympäristössä esim. teknisesti suojatun turva-alueen sisäisissä kaapeloinneissa, tai hallintatyöasema kytketään matalamman tason salauksella laitteeseen. Matalamman tason salauksena voidaan käyttää esim. SSH, HTTPS ja SCP suojattuja yhteyksiä. (KATAKRI 2015a, 35–36). (Viestintävirasto 2015b).

3.2.2 Käyttäjän todentaminen

Suojaustasolla IV vaatimuksena on, että jokaisella käyttäjällä on henkilökohtaiset käyttäjätunnukset, ja jokainen käyttäjä tunnistetaan ja todennetaan. Tunnistaminen ja todentaminen tulee olla järjestetty käyttäen turvallista ja tunnettua tekniikka, tai muuten luotettavasti järjestetty. Liian monta kertaa peräkkäin epäonnistunut tunnistus lukitsee käyttäjätilin. Ylläpitotunnukset ovat henkilökohtaisia, jos käytössä on ylläpitoon yhteyskäyttötunnukset, vaaditaan niistä sovittu ja dokumentoitu salasanojen hallintakäytäntö. Todentaminen tehdään käyttäen vähintään salasanaa, josta käyttäjää on oheistettu salasanan käytöstä ja valinnasta hyvän turvallisuuskäytännön mukaisesti. Salasanan käyttöä valvovan ohjelmiston tulee asettaa salasanan vaihdolle

sopivat määräajat, sekä asettaa salasanalle tietyt turvallisuuden vähittäisvaatimukset. (KATAKRI 2015a, 40).

Ympäristö jossa palvelunestohyökkäyksen uhka arvioidaan merkittäväksi esim. Internet palveluissa tapahtuva todentaminen, voidaan tunnuksien lukittuminen korvata muulla riskiä vähentävällä periaatteella. Tunnuksen todentamisessa voidaan käyttää viivettä, suodattamista tai tunnuksen väliaikaista lukittumista. Päätelaitteen teknistä tunnistamista ei vaadita suojaustasolla IV, jos käyttäjä tunnistetaan. (KATAKRI 2015a, 41).

Suojaustasolla III ja II vaatimuksena on suojaustason IV vaatimusten lisäksi vahva käyttäjätunnistus, joka perustuu vähintään kahteen tekijään. Teknisen päätelaitteen tunnistaminen ennen laitteen pääsyä verkkoon tai palveluun käyttäen laitetunnistusta, 802.1X, tai muuta vastaavaa menettelyä. Jos päätelaitteen verkkoon kytkeytyminen on fyysisesti rajattu suppeaksi, esim. viranomaisen hyväksymän ko. suojaustason laitekaappi, ei päätelaitteen tunnistaminen ole välttämätöntä. (KATAKRI 2015a, 41).

Tiukasti fyysisesti rajatulta suoja-alueelta joka on teknisesti suojattu turva-alue, lukittu laitekaappi, tai vastaava, ja jonka pääsynvalvonnassa käytetään kahteen tekijään perustuvaa vahvaa tunnistamista. Tällöin voidaan joissain tapauksissa käyttäjän tunnistamiseen käyttää käyttäjätunnus-salasana menetelmää suojaustason III ja II tietojärjestelmässä. (KATAKRI 2015a, 41).

Todentamisessa ja tunnistamisessa tulee huomioida, että sisäänkirjautuessa ennen todentamista ei paljasteta tarpeetonta tietoa, todennusmenetelmät ovat suojattu välimieshyökkäyksiltä, tunnistamistiedot todennuksessa ovat salatussa muodossa jos ne lähetetään verkon yli, todentamiseen käytettävät menetelmät ovat suojattu uudelleenhyökkäyksiltä, ja todentaminen on suojattu brute force-hyökkäyksiltä. (KATAKRI 2015a, 41).

3.2.3 Verkkolaitteen koventaminen

Verkon aktiivilaitteilla tarkoitetaan verkossa olevia langattomia tukiasemia, reitittimiä, kytkimiä, palomuureja ja vastaavia laitteita. Järjestelmillä tarkoitetaan palveluita

tukevia ja tuottavia laitteita. Järjestelmiä ovat työasemat, palvelimet, verkon laitteet ja vastaavat. Järjestelmän koventamisella tarkoitetaan asetuksien ja palveluiden muuttamista niin että niiden haavoittuvuuspinta-ala on mahdollisemman suppea. Järjestelmien ja palveluiden kannalta otetaan käyttöön vain toiminnollisuuksien kannalta oleellisimmat asetukset ja palvelut. Järjestelmien automaattisille prosesseille annetaan vain ne valtuudet, tiedot ja oikeudet, jotka ovat prosesseille välttämättömiä. Toiminnollisuuksia ja asetuksia rajoittamalla vähennetään virheistä, onnettomuuksista, ja järjestelmän resurssien luvattomasta käytöstä aiheutuvia riskejä ja vahinkoja. Työasemien ja palvelinten riittävän kovennuksen voi toteuttaa USGCB:n tai vastaavan avulla. Jos salassa pidettävän tiedon käsittelyssä käytetään muita laitteita esim. verkkotulostimia ja puhelimia, tulisi niissä myös soveltaa edellä mainittuja järjestelmän koventamiseen liittyviä periaatteita. (KATAKRI 2015a, 42).

Verkonlaitteen hallintaan suuressa ympäristössä voidaan käyttäjän todennukseen käyttää kahdennettuja AAA-palveluita, esim. Kerberos, TACACS+, tai RADIUS. Jos verkkolaitteen hallinta yksilöivällä käyttäjätunnuksella ei ole mahdollista, voidaan tunnistaminen suorittaa hyväksi todetulla käyttösäännöllä, esim. salasanaan pääsy vaatii kahden henkilön osallistumista. (KATAKRI 2015a, 42).

Verkon aktiivilaitteille on suojaustason IV vaatimuksena, oletussalasanojen vaihto organisaation salasanapolitiikan mukaisiin salasanoihin, verkkopalvelut rajoitettu vain tarpeellisiin verkkoliittymiin, verkkolaitteisiin on asennettu tarpeelliset turvapäivitykset, verkkolaitteen hallinta ei ole mahdollista ilman käyttäjän tunnistamista ja todentamista, istuntojen aikakatkaisu hallintayhteyksissä, ja verkkolaitteen koventaminen pohjautuu luotettavaan kovennusohjeeseen ja suositukseen. (KATAKRI 2015a, 42).

Verkon laitteille suojaustason III ja II vaatimuksina tulee edellä mainittujen lisäksi poistaa laitteista käytöstä tarpeettomat loogiset ja fyysiset liityntärajapinnat. (KATAKRI 2015a, 43).

Suojaustason IV järjestelmissä vaatimuksena on verkkopalveluiden rajaaminen vain välttämättömiin palveluihin, palvelut tulee rajata palomuurilla vain välttämättömään liikenteeseen. Järjestelmän alusta sisältää vain toiminnollisuuksien ja palveluiden kannalta välttämättömiä ohjelmistokomponentteja. Käyttöoikeudet ohjelmistokom-

ponenteille, palvelinprosesseille, hakemistoille ja lisäohjelmille tulee toteuttaa vähimpien oikeuksien periaatteen mukaisesti. Sovellusohjelmistoilla ja käyttöjärjestelmillä tulee olla asennettuna ajantasaiset ja tarpeelliset tietoturvapäivitykset. Järjestelmissä ei käytetä oletustilejä esim. guest- ja administrator. Jos oletustilejä ei ole mahdollista poistaa, tulee niiden oikeudet rajata mahdollisimman alhaisiksi. Oletussalasanoja ei käytetä, ja ne on muutettu organisaation salasanapolitiikan mukaisiksi. Järjestelmän tulee lukittua automaattisesti jos sitä ei käytetä hetkeen. Käyttöoikeudet tulee asettaa vähimpien oikeusien periaatteen mukaisesti. Tunnettujen käyttöjärjestelmän turvallisuusuhkien estämiseksi automaattisten ohjelmakoodien suorittaminen on kytkettävä pois päältä, esim. USB-laitteiden automaattinen käynnistyminen on estetty käyttöjärjestelmän ollessa lukittuna. Työpöytäohjelmistot ja web-selaimet turvallisesti konfiguroituja, ja niistä on oletuksena poistettu käytöstä tarpeettomat liitännäiset ja laajennustyökalut, erityisesti tulee huomioida ajettavat koodit esim. JavaScript, joka tulisi estää oletuksena. Järjestelmän BIOS-asetuksiin pääsy tulee suojata salasanalla organisaation salasanpolitiikan mukaisesti. Järjestelmissä hyödynnetään niitä tukevia lisäturvallisuusosia, esim. SELINUX, Applocker, DEP, ASLR, tai vastaavia. (KATAKRI 2015a, 43).

Järjestelmille suojaustason III ja II vaatimuksina tulee edellä mainittujen lisäksi poistaa käytöstä tarpeeton verkkoliikennöinti. Järjestelmien ohjelmistot asetetaan hakemaan päivitykset vain niille tarkoitetuista lähteistä. BIOS-asetuksien muuttaminen estetty valtuuttamattomilta käyttäjiltä, käynnistys sallitaan vain ensisijaiselta kovalevyltä, ja tarpeettomat palvelut ja portit poistetaan käytöstä. (KATAKRI 2015a, 43).

3.2.4 Jäljitettävyys

Tietojenkäsittely-ympäristössä tietojen luvaton muuttaminen ja asiattomien tietojen käsittely havainnointi tulee olla jäljitettävissä. Palvelinten, työasemien ja verkkolaitteiden keskeisiä tallenteita ovat kirjautumis- ja lokitiedot. Järjestelmien kattavuusvaatimus voidaan toteuttaa vastaavien lokitietojen päällä pitämisellä. Verkkolaitteiden lokitiedoista tulee myös selvitä mitä laitteelle on tehty, milloin on tehty, ja kenen toimesta asetuksia on muutettu tai selailtu. Verkkolaitteista ja järjestelmistä tapatu-

malokeja tulisi kerätä turvallisuuteen liittyvistä toiminnollisuuksista, kirjautumistiedoista, sekä järjestelmästä poikkeavista tapahtumista. Lokitiedot tulisi tallentaa keskitettyyn paikkaan vahvasti suojatulle palvelimelle, jonka tietoja varmuuskopioidaan säännöllisesti. (KATAKRI 2015a, 46).

Useissa järjestelmissä lokitus ei ole oletusarvoisesti päällä, tämä vaatii lokituksen päälle laittamista ja sen oletusarvojen muuttamista lokitietojen säilytysajan ja tilan suhteen. Windows työasemissa ja palvelimissa valvontakäytäntöjä muuttamalla voidaan lokittaa epäonnistuneita ja onnistuneita tapahtumia, tilienhallinnasta- ja kirjautumistapahtumista, käytäntöjen muutoksista ja järjestelmätapahtumista. Järjestelmissä tulee myös ottaa huomioon lokitietojen vaatima säilytystila ja aika. Suositeltava aika lokien säilyttämiselle on esimerkiksi arvioida yhden kuukauden lokitietojen kertymän vaatima säilytystila. Säilytys tilaa tulisi olla reilusti arvioitua enemmän, poikkeavissa tilanteissa lokitietoa kertyy huomattavasti enemmän ja erilaiset hyökkäystyypit voivat kasvattaa lokitietojen määrää huomattavasti. (KATAKRI 2015a, 46).

Suojaustason IV järjestelmissä vaatimuksena on että tallenteita on riittävä määrä tietomurtoyritysten ja niiden jälkikäteiseen todentamiseen, kriittiset tallenteet säilytetään vähintään 6kk, lokitiedot suojataan luvattomalta pääsyltä, organisaatiolla on kirjallinen ohje lokitietojen seurantapolitiikalle ottaen huomioon organisaation toiminnan vaatimukset. (KATAKRI 2015a, 46).

Suojaustasolle III ja II on edellä mainittujen lisäksi vaatimuksena, kriittisten lokitietojen säilyttäminen vähintään 2-5 vuotta, säännöllisin väliajoin lokitietojen varmuuskopiointi, saman turvallisuusalueen sisäiset tietojärjestelmät synkronoidaan sovitun ajanlähteen kanssa, lokitietojen eheys varmistetaan, ja lokitietojen käsittelystä ja käytöstä muodostetaan merkintöjä. (KATAKRI 2015, 46).

3.2.5 Poikkeavuuksien havainnointi

Tietojärjestelmä-ympäristössä on toteutettu menetelmät, joilla pyritään havaitsemaan ja estämään mahdolliset hyökkäykset, vahingonteot, rajaamaan vaikutukset

mahdollisemman suppeaksi, sekä palauttamaan mahdollisesta hyökkäyksestä takaisin suojattuun tilanteeseen. (KATAKRI 2015a, 48).

Tietoverkkoliikennöinnin tarkkailuun on olemassa monia soveltuvia menetelmiä, aina solmupisteiden tasolla tapahtuvasta liikenteen tarkkailusta työasema- ja palvelinkohtaisiin sensorijärjestelmiin ja näiden yhdistelmiin. Riippumatta käytettävästä menetelmästä ja laitevalmistajasta, tietoverkkoliikenteen havainnointikyky edellyttää verkkoliikenteen normaalin tilan tuntemista. Suojaustasolle IV riittävä verkon havainnointikyky voidaan toteuttaa verkon ulkoreunan solmupisteellä, suojaustasosta III lähtien verkon havainnointikyky tulee sisältää yhdyskäytäväratkaisut ja verkon sisäliikenteen. Hyökkäysten havainnointi edellyttää automatisoituja havainnointi- ja hälytystyökalujen käyttöä. Ympäristö jossa lokitietojen määrä on suppea, voidaan riittävällä henkilöstöresurssilla toteuttaa havainnointi perustuen lokitietoihin ja raportteihin. Suojattuun tilaan palautuminen vaatii tietojenkäsittely-ympäristössä hyvin suunniteltuja, koulutettuja ja harjoiteltuja prosesseja sekä menetelmiä. (KATAKRI 2015a, 48).

Suojaustasolle IV-II vaatimuksena on tuntea verkkoliikenteen normaali tila, sen käytetyistä protokollista, liikennemääristä ja yhteyksistä. Kohde organisaatiolla tulee olla menettely poikkeavan verkkoliikenteen havaitsemiseen, sekä tallenteita poikkeavuuksien todentamiseen jälkikäteen. Organisaatiolla tulee myös olla prosessoitu menetelmä poikkeavuuksista toipumiseen. (KATAKRI 2015a, 48).

3.2.6 Sähköisen aineiston välitys fyysisesti suojatun alueen sisällä

Salassa pidettävä materiaalin liikkuessa fyysisesti suojatun alueen ulkopuolelle, verkkoliikenne/materiaali tulee salata ko. suojaustasolle viranomaisen hyväksymällä menetelmällä. Salassa pidettävän materiaalin siirtäminen fyysisesti suojatun alueen sisällä, voidaan verkkoliikenteen/materiaalin siirtoon käyttää alemman tason salausta. (KATAKRI 2015, 53).

Tietoverkossa tiedonsiirtoon voidaan käyttää puhelimia, telekopiota, sähköpostia ja muita pikaviestimiä. Radiorajapinnan käyttö fyysisesti suojatulla alueella tulkitaan

suojatulta alueelta poistumiseksi. Langattomien verkkojen radiorajapintaa tulisi aina käsitellä kuten julkista tiedonsiirtoverkkoa. (KATAKRI 2015a, 53).

Suojaustasoille IV-II vaatimuksena on, siirrettäessä fyysisesti suojatun alueen ulkopuolelle salassa pidettävää materiaalia tulee materiaalin ja verkkoliikenteen salaamiseen ko. suojaustasolle käyttää viranomaisen hyväksymää salausmenetelmää. Tilanteet jossa salassa pidettävää materiaalia siirretään suojatun alueen sisällä, tulee ko. suojaustason liikennekanava olla fyysisesti suojattu, tai materiaali/aineisto voidaan suojata matalamman tason suojauksella, esim. HTTPS. (KATAKRI 2015a, 53).

3.2.7 Etäkäyttö ja – hallinta

Etäkäytöllä ja etähallinnalla tarkoitetaan organisaation ulkopuolelta tapahtuvaa tietojärjestelmän hallinnointia, siihen tarkoitetulla päätelaitteella. Päätelaitteena toimii normaalisti kannettava tietokone, perinteisessä merkityksessä etähallintaa voidaan toteuttaa vain suojaustason IV tietojärjestelmäympäristöllä. Suojaustasosta III lähtien materiaalin/aineiston käsittely edellyttää viranomaisen hyväksymää suoja-aluetta. Viranomainen voi hyväksyä korvaavan menettelyn, joilla saavutetaan vastaavat fyysisen turvallisuuden olosuhteet, esim. viranomaisoperaatiot. (KATAKRI 2015a, 62).

Etäkäytön- ja hallinnan vaatimuksena suojaustasolle IV on, suojattujen alueiden välillä tapahtuva materiaalin välitys ja käsittely ko. suojaustasolla on mahdollista vain viranomaisen hyväksymien korvaavien menettelyiden mukaisesti. Tällaisia korvaavia menettelyjä ovat, vain käyttöympäristöön hyväksytyt laitteet ja järjestelmien etäkäyttöön edellyttävää vahvaa kahteen tekijään perustuvaa tunnustautumista. Organisaation etäkäyttöä- ja hallintaa toteuttavat henkilöt tulee olla koulutettu ja ohjeistettu organisaation tietoturvallisuusperiaatteisiin. Suojatun alueen ulkopuolelle viedyt suojaustason IV tietovälineet, esim. USB-muistit, kiintolevyt, tulee salata virnaomaisen ko. suojaustasolle hyväksymällä salausmenetelmällä. Salassa pidettävä materiaali ei jätetä suojaamattomalla alueella valvomatta, materiaalin voi säilyttää vastalle tasolle hyväksytyllä lukittavassa toimistotilassa. Tietojärjestelmien etäkäyttö- ja hallintaratkaisut vaatii viranomaisen ko. suojaustasolle hyväksymää liikenteen salausmenetelmää. (KATAKRI 2015a, 62).

Suojaustasolle III ja II aiemmin mainittujen vaatimusten lisäksi tulee, fyysisesti suojatun alueen ulkopuolelle vietyä materiaalia/aineistoa ei saa avata eikä lukea julkisella paikalla, materiaalin tulee aina olla sen kuljettajan hallussa, ellei materiaalia/aineistoa ole salattu viranomaisen ko. suojaustasolle hyväksytyllä salausmenettelyllä. Tietojärjestelmien etäkäyttö- ja hallinta rajoitetaan viranomaisen hyväksymälle fyysisesti suojatun alueen sisälle. (KATAKRI 2015a, 62).

3.2.8 Varmuuskopiointi

Varmuuskopioinnilla tarkoitetaan alkuperäisen tiedon tallentamisesta toissijaiselle tallennuslähteelle esim. nauhatallenne, josta alkuperäinen tieto voidaan palauttaa/hakea tarpeen mukaan. (KATAKRI 2015a, 65).

Varmuuskopiot jotka sisältävät salassa pidettävää tietoa, tulisi säilyttää saman tasoisilla menetelmillä kuin alkuperäinenkin tieto. Varmuuskopiot tulisi mitoittaa toimintavaatimuksia vastaavaan tasoon. Tiedon kriittisyyteen nähden, tulisi ottaa huomioon varmistusten taajuus, edellytyksenä selvitys siitä kuinka paljon tietoa voidaan menettää. Palautusprosessi tulee mitoittaa palautettavan järjestelmän kriittisyyden mukaan, huomioiden kuinka kauan järjestelmän palautuminen kestää. Varmuuskopiointi ja palautusprosessi tulisi testata säännöllisesti oikean toiminnollisuuden varmistamiseksi. Varmuuskopioiden fyysinen sijoituspaikka tulisi olla riittävän hyvin eriytetty varmuuskopioitavasta lähteestä, esim. sortuma- ja palovaaran takia. Samalla varmistusjärjestelmällä tallennettujen eri omistajien tietueisiin tulee olla erottelumenettely, esim. omistajakohtaiset materiaalit/tiedot salataan omalla salausavaimella, joita säilytetään asiakaskohtaisissa kassakaapeissa. (KATAKRI 2015a, 65).

Suojaustasosta IV lähtien varmuuskopiot säilytetään koko niiden elinkaaren ajan saman tasoisessa järjestelmässä. Varmistusmediat tulee hävittää ko. suojaustason mukaisesti. Lisäksi suojaustasolla III ja II varmuuskopioista pidetään rekisteriä, joihin kirjataan varmuuskopioiden käsittelystä merkintä sähköiseen lokiin, asianhallintajärjestelmään, tietojärjestelmään ja manuaaliseen diaariin tai asiakirjaan. (KATAKRI 2015a, 65).

4 Automatisointia tukevat tekniikat

4.1 TCL

4.1.1 Yleistä

Tcl:n (Tool Command Language) on keksinyt John K. Ousterhout 1980 luvun loppupuolella Kalifornian yliopistossa Berkleyssä. Tcl on dynaaminen ohjelmointikieli, jota voidaan käyttää skriptien luontiin, tulkkikielenä ja C-kirjastona. Tcl auttaa hallitsemaan ja komentamaan muita ajettavia ohjelmia ja toimintoja ohjelmassa. Tcl on tulkittu ohjelmointikieli. Tulkitun ohjelmointikielen etuna käännettyyn ohjelmointikieleen on sen nopea kehitysprosessi; skriptiin voidaan tehdä muutoksia nopeasti ja suorittaa niitä nähdäkseen muutoksen tuomat vaikutukset. Etuna on myös skriptien luominen tekstimuodossa. Tulkkikielen haittapuolena on sen suorituskyky. Suorituskyky riippuu ajettavan alustan käyttöjärjestelmästä, prosessorista ja ohjelmointikielestä. Tcl-skripti täytyy ensiksi tulkita, ennen kuin se suoritetaan. Toinen haittapuoli on skriptikoodin piilottaminen. Koska skriptit tehdään tekstimuodossa, voi kuka tahansa, joka saa skriptin käyttöönsä, muokata ja kopioida sitä. Suorituskyvyn lisäksi tulkattavissa ohjelmointikielissä on haittapuolena niiden muistin vaatimus. Koko skripti, sen tulkattu muoto ja muuttujat pidetään välimuistissa. (Blair 2010).

Tcl skriptaamisen avainedut ovat tietojen tulostaminen ja hakeminen muista laitteista, käyttöliittymistä ja tietokannoista. Skripteillä voidaan automatisoida monimutkaisia tehtäviä. Informaatioita voidaan muokata erilaisilla kokonaisluvuilla ja muuttujilla. Tcl on myös helposti opittava tulkki kieli. (Blair 2010).

Tcl-kieleen kuuluu myös useita eri komponentteja, joista tunnetuin on Tk (Tool Kit) komponentti. Tk-komponentin avulla ohjelmalle voidaan luoda graafinen käyttöliittymä. Komponentilla voidaan luoda ikkunoita, painikkeita, tekstilaatikoita jne. Tcl-kieltä käytetään yleisesti ohjelmien testaamiseen, automatisointiin, web-käyttöliittymissä, työpöytäohjelmissa, tietokannoissa ja sulautetussa ympäristöissä. Tcl-kieli on

saavuttanut suosiota helppona ohjelmointikielenä, jota voi ajaa lähes kaikilla alustoilla. Se eroaa muista skriptikielistä siinä, että sen voi helposti sulauttaa muihin sovelluksiin. Lisäksi Tcl on ilmainen ja Tcl-yhteisön ylläpitämä. (Blair 2010).

4.1.2 Tcl kieli

Tcl-skriptit on tehty komennoista, joita erotellaan rivinvaihdoilla ja puolipisteillä. Kaikki komennot ovat rakenteeltaan samanlaisia.

expr 20 + 10

Tällä komennolla lasketaan luvun 20 ja 10 yhteenlasku, josta saadaan tuloksena 30. Jokaisessa Tcl komennossa on yksi tai useampi sana, nämä sanat on eroteltu välilyönneillä. Yllä olevassa esimerkissä on neljä sanaa: expr, 20, + ,10. Ensimmäinen sana on komennon nimi, muut sanat ovat komennon argumentteja. Kaikki komennot koostuvat sanoista, mutta eri komennot käsittelevät argumentteja eri tavalla. (Tcl Developper 2015a).

Muuttujat

Tcl kielessä voi tallentaa arvoja muuttujiin joita voidaan hyödyntää myöhemmissä komennoissa. *Set*-komennolla tallennetaan ja luetaan muuttujia.

set x 32

Yllä olevassa esimerkissä komennolla *set* määritellään muuttujalle *x* arvoksi *32*.

set x

Muuttujan *x* arvo voidaan yksinkertaisesti lukea komennolla *set x*. Muuttujien arvoa ei tarvitse määrittää jokaisen komennon yhteydessä. Muuttujan arvo säilyy, kun se on ensimmäisen kerran määritelty.

*expr $x*3*

Kun komennossa ilmenee *$*, Tcl käsittelee sen jälkeisiä numeroita, kirjaimia ja sanoja muuttujien nimenä. Aiemmassa esimerkissä muuttujalle *x* asetettiin arvo 32. Tässä esimerkissä oikea lasku on siis 32*3. (Tcl Developper 2015a).

Komentojen korvaaminen

Komennon tulosta voidaan myös käyttää muuttujan arvona.

```
set a 44
set b [expr $a*4]
```

Komennossa ilmenee hakasulkeet [], Tcl käsittelee kaiken hakasulkeiden sisällä olevat komennot sisäkkäisesti. Esimerkissä Tcl käsittelee hakasulkeiden sisällä olevat komennot, ja laskee niiden tuloksen. Muuttujalle *b* saadaan arvoksi 176.

Lainausmerkit ja aaltosulkeet

Lainausmerkeillä voidaan määrittää muuttujien arvoja, jotka sisältävät välilyöntejä.

```
set x 24
set y 18
set z "$x + $y is [expr $x + $y]"
```

Kun nämä kolme komentoa on suoritettu, muuttujalla *z* on arvo *24 + 18 is 42*. Kaikki lainausmerkkien sisällä lähetetään *set* komennolle yhtenä sanana. Jos komennossa ei ole lainausmerkkejä, *set* komento saa kuusi argumenttia, jotka aiheuttavat virheen.

Aaltosulkeilla voidaan myös määrittää muuttujien arvoja sanoiksi, jotka sisältävät välilyöntejä. Aaltosulkeet eroavat lainausmerkeistä siinä, että komentoa tai laskutoimitusta ei tehdä aaltosulkeiden sisällä.

```
set z {$x + $y is [expr $x + $y]}
```

Komennolla saadaan muuttujalle *z* arvoksi $x + $y is [expr $x + $y]. (Tcl Developper 2015a).

PROC -Komentorakenne

Tcl kielen avulla voidaan luoda useita eri komentorakenteita eri proseduureille, jotka käsittelevät Tcl skriptiä argumenttina. Esimerkissä luodaan Tcl proseduuri *value*.

```
proc value {base p} {
        set result 1
        while {$p > 0} {
                set result [expr $result * $base]
                set p [expr $p – 1]
                }
        return $result
        }
```

Esimerkissä käytetään vain yhtä komentoa *proc*. Komennolla *proc* on kolme lausetta, proseduurin nimi, lista argumenttien nimistä, sekä proseduurin rakenne joka on Tcl skripti. Aaltosulkeiden sisällä olevat argumentit jotka ovat alussa ja lopussa, lähetetään sanatarkkana *proc* komennolle yhtenä argumenttina. *Proc* komento luo uuden Tcl komennon nimeltä *value*, jolla on kaksi argumenttia. *Value* komentoon voidaan vedota esimerkiksi:

```
value 2 6
value 1.15 5
```

Kun *value* komentoon on vedottu arvoilla, käynnistää se proseduurin. Kun proseduuri on käynnissä, se käyttää annettuja arvoja argumentissa, jossa *Base* saa ensimmäisen arvon, ja *p* saa toisen arvon.

Value komennon proseduuri sisältää kolme Tcl lausetta, *set*, *while*, *return*. *While* lause tekee suurimman osan proseduurista. Se saa kaksi argumenttia, lausekkeen (*$p*

> 0) ja rakenteen, joka on toinen Tcl skripti. *While* lause arvioi sen lausekkeen argumentin arvon käyttäen samanlaisia sääntöjä kuten C-ohjelmointikielessä, jos arvo on tosi (ei nolla), se toteuttaa rakenteen. Se toteuttaa prosessin niin pitkään kunnes se saa arvon epätosi (arvo nolla). Tässä tapauksessa *while* komento kertaa argumentin *base* arvon, ja vähentää *p* argumentin arvoa. Kun *p* saa arvoksi epätosi (nolla), on tulos haluttu arvo lauseen argumentille *base*. *Return* komento lopettaa proseduurin ja palauttaa muuttujalle arvon muuttujasta *result*, joka on proseduurin tulos. (Tcl Developper 2015a).

IF – Komentorakenne

If-komennolla voidaan tutkia onko annettu arvo tosi, vai epätosi. *If*-komento arvioi annettuja arvoja ilmaisuna, *true* eli tosi, ja *false* eli epätosi. Ilmaisun arvo voi olla boolean, arvolla *0* tulos on epätosi, ja kaikilla muilla arvoilla *N* tulos on tosi.

if value1 ?then? body1 elseif value2 ?then? body2 elseif … ?else? ?bodyN?

Esimerkissä *if* -komento arvioi *value1* arvoa, joka on boolean arvo. Jos arvo on tosi, lähetetään se TCL komentotulkille ja suoritetaan *body1*-komento. Muussa tapauksessa arvioidaan arvoa *value2*, jos se on tosi, suoritetaan *body2*-komento. Jos mikään edellä annetuista arvoista ei ole tosi, suoritetaan *bodyN*-komento. *Then*, *else* ja *elseif* ovat tosi arvoilla suoritettavia lauseita, joita voi olla *N* määrä. *If*-komennon tulokseksi saadaan koko suoritetun komentoketjun arvo. Arvo voi olla myös nolla, jos mikään arvo ei toteuttanut *body*-komentoja, ja lopussa ei ole *bodyN*-komentoa.

```
if {$value == 1} {
                puts ”value on 1”}
 elseif {$value == 2} {
                puts “value on 2”}
 else {
                puts “ value ei ole 1 tai 2”

}
```

Esimerkissä verrataan *if*-komennolle annettua arvoa *$value*. Jos arvo on yksi, tulostetaan *”value on 1”*, jos arvo on kaksi, tulostetaan ”value on 2”. Jos arvo ei ole kumpikaan edellä mainituista tulostetaan *”value ei ole 1 tai 2”*. *If*-komennossa arvona voidaan käyttää useita arvoja samanaikaisesti, kun käytetään useita arvoja, voidaan lisätä *then*-lause. (Tcl Developper 2015c).

```
if { $value == 1 | |$value == 2 | |$value == 2
                } then {puts “value on yksi, kaksi tai kolme”}
```

FOREACH-Komentorakenne

Foreach on silmukkakomentoa, johon voidaan luoda lista muuttujista. Yksinkertaisimmillaan silmukkakomento sisältää yhden muuttujan *varN*, ja yhden listan *listN*. Listassa voi olla useita arvoja, jotka kaikki ovat sidottu samaan muuttujaan. Saman *foreach*-komentorakenteen alle voidaan luoda useita listoja erilaisilla muuttujilla. *Body* on TCL-skripti joka suoritetaan kerran jokaiselle silmukkakomennon toistolle.

```
foreach varN listN body

foreach var1 list1 ?var2 list2 … varN listN? body
```

Jokaista listan argumenttia voidaan muokata *body*-lauseessa. *Foreach*-komento ei tunnista argumenttien muokkausta, kun silmukan toistoa suoritetaan. TCL-komentotulkki luo kopion listasta, jotta silmukkakomennolle ei tule päättymätöntä toistoa. Jos lista on hyvin suuri, voi silmukkakomento jatkaa toistoa kunnes järjestelmän muisti tulee täyteen.

```
foreach a [list 1 2 3 4] b [list 5 6 7 8] c [list a b c d] d [list w x y z]
                {puts “$a $b $c $d”}
```

Esimerkissä luodaan neljä listaa *a, b, c* ja *d*. Jokaisen listan muuttujaa kutsutaan kerran järjestyksessä vasemmalta oikealle. *Puts*-komennolla tulostetaan jokaisesta listasta arvot järjestyksessä. Tuloksena saadaan kaikkien listojen arvot tulostettuna.

1 5 a w

2 6 b x

3 7 c y

4 8 d z

Jos listassa ei ole riittävästi arvoja jokaiselle silmukan muuttujalle, käytetään tyhjiä arvoja muuttujan täydentämiseen. *Foreach*-komentoa voidaan käyttää, kun halutaan *ping*-komennolla testata saatavuus useisiin muihin verkossa oleviin laitteisiin samanaikaisesti. (Tcl Developper 2015b).

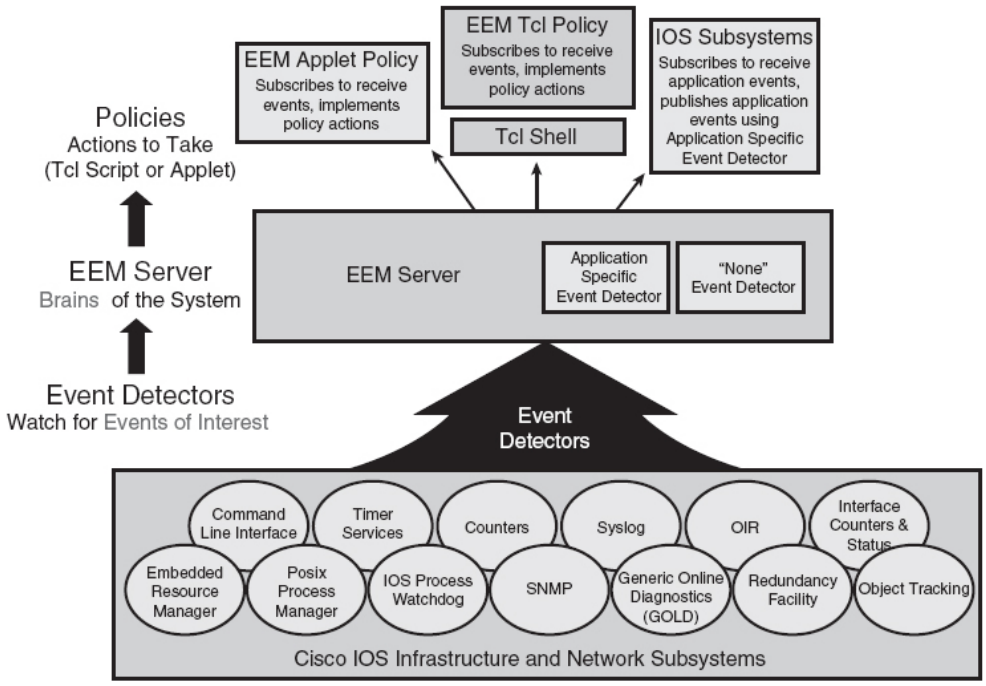
4.1.3 Tcl ja Cisco IOS

Ciscon IOS käyttöjärjestelmissä Tcl tuki esitettiin ensimmäisen kerran versiossa 12.3(2)T ja 12.2(25)S. Tcl tuki tuli osaksi Cisco IOS käyttöjärjestelmää Catalyst 6500 sarjan modulaarisissa kytkimissä versiossa 12.2(18)SX4. IOS käyttöjärjestelmistä Tcl skripti voidaan ajaa omasta käyttöliittymästä. Käyttöliittymään pääsee käsiksi komennolla *tclsh*, komennon voi suorittaa vain *EXEC* tilassa. Cisco IOS käyttöjärjestelmässä useat aliohjelmat tukevat Tcl skriptejä, tällaisia aliohjelmia ovat mm. ESM, EMM, IVR ja EEM. (Blair 2010).

4.2 EEM

Cisco *Embedded Event Manager* (EEM) on Ciscon IOS järjestelmissä oleva toiminnollisuus, jonka avulla voidaan Cisco IOS laitteissa suorittaa käyttäjän tekemiä ohjelmallisia ja automaattisia toimintoja herätteen saadessa, tai tiettynä ajankohtana. EEM toiminnollisuuden avulla käyttäjä voi luoda skriptejä, valvoa järjestelmän kynnysarvoja, tehdä ennakoivia toimenpiteitä, suorittaa tiedon keruuta, ja valvoa tapahtumia Cisco IOS laitteessa, ilman erillistä palvelinta. EEM avulla tapahtumia ja ongelmia voidaan etukäteen automaattisesti ratkaista valvomalla laitteen luomia herätteitä (Event Detectors), järjestelmä voidaan asettaa suorittamaan haluttuja toimintoja myös ajastetusti. Tapahtuman tai herätteen toteuduttuaan, järjestelmä suorittaa tapahtumalle ennalta asetetun politiikan mukaiset toiminnot (ks. Kuvio 2). EEM koostuu kolmesta

osa-alueesta, herätteiden valvonnasta (Event Detector), ennalta asetetuista toiminnoista (Policies), ja EEM palvelimesta (EEM Server). (Blair 2010).



Kuvio 2. EEM monitorointi (Blair 2010)

4.2.1 Herätteiden valvonta

Herätteiden valvonta monitoroi Cisco IOS laitteen prosesseja, ja toiminnollisuuksia. Herätteen saadessaan Event Detector lähettää hälytyksen EEM palvelimelle, joka käynnistää herätteen politiikan mukaisen toiminnollisuuden. Heräte sisältää tietoa herätteen tyypistä ja ajankohdasta. Tyypillisimpiä herätteitä ovat SYSLOG viestit, SNMP herätteet, ajastetut toiminnot, ja verkkorajapinta muutokset. Herätteitä voidaan kerätä samanaikaisesti useasta prosessista, herätteen tyypistä riippumatta. (Blair 2010).

4.2.2 Politiikka

Ennalta asetetut toiminnot voivat olla Cisco IOS appletteja tai Tcl skripteistä. Appletteja luotaessa käyttäjän ei tarvitse osata Tcl ohjelmointikieltä, appletit koostuvat Cisco IOS komennoista. Applettien asetukset näkyvät Cisco IOS laitteiden asetuksissa,

Tcl skriptien sisältöä ei näe laitteen asetuksista suoraan. Tcl skriptit tallennetaan laitteen omaan paikalliseen muistiin, joka on tyypillisesti flash-muisti. Laite voi myös itse hakea Tcl skripti verkossa olevalta palvelimelta tai työasemalta, jossa on tiedostojen jakaminen mahdollistettu. (Blair 2010).

4.2.3 EEM Palvelin

EEM palvelin (EEM Server) toimii siltana Cisco IOS aliohjelmien luomien herätteiden (Event Detector), ja ennalta asetettujen toimintojen (Policies) välillä. EEM palvelimen tehtävänä on rekisteröidä Cisco IOS aliohjelmissa tapahtuvia herätteitä, varastoida tietoa herätteestä, julkaista tapahtuma herätteestä, pyytää lisää tietoa herätteestä, rekisteröidä sisäiset ohjelmalliset hakemistot, rekisteröidä Tcl skriptit ja sovellukset, ja prosessoida käyttäjän asettamat toiminnot herätteille. (Blair 2010).

5 TOTEUTUS

5.1 Yleistä

Toteutuksessa keskitytään luomaan Cisco IOS järjestelmiin automaattisia ja yhtenäisiä toimintoja. Yhtenäisillä toiminnoilla pyritään varmistamaan usean eri käyttäjän tekemien muutosten yhtenäisyys verkonlaitteissa, tällä pyritään välttämään mahdolliset ristiriitaisuuden verkkolaitteiden asetuksien välillä. Toteutuksessa huomioidaan myös KATAKRI:n asettamia vaatimuksia eri suojaustasoilla tietotojen käsittelyyn ja tiedon siirtämiseen toteutusympäristössä. Toteutusympäristössä otetaan huomioon, se että kaikissa verkkoympäristöissä ei ole mahdollista hallintapalvelinta laitteille ja lokipalvelinta tapahtumien keräämiseen laitteista.

5.2 Reitittimen asetukset skriptillä

Yleisenä ongelmana reitittimen tai muun verkkolaitteen käyttöönotossa on alkuasetuksien asettaminen yhtenäisesti. Kun verkon ylläpitäjiä on useita, vaihtelevat

käytännöt verkkolaitteen käyttöönotostakin yhtä paljon. Moni ylläpitäjä voi pitää itsellään työasemalla muistilistaa tai tiedostoa, josta löytyy verkkolaitteen perus asetukset. Tällainen ylläpito voi olla työlästä, koska verkkolaitteet voivat vaihdella tyypeittäin ja kaikissa laitteissa ei ole yhtenevät järjestelmäpäivitykset.

Tätä toimintoa voi helpottaa tekemällä tarkoitukseen sopivan Tcl-skriptin. Skriptillä voidaan automatisoidusti asettaa tarvittavat asetukset verkkolaitteeseen. Skripti myös helpottaa ylläpitäjää muistamaan, mitä asetuksia tulee verkkolaitteelle asettaa.

Toteutuksessa otetaan huomioon KATAKRIn asettamia vaatimuksia verkkolaitteen koventamiselle, kun laitteesta poistetaan turhia palveluita käytöstä. Hallintayhteyksissä huomioidaan KATAKRIn vaatimus tietoturvalliselle hallintayhteydelle.

Kuviosta 3 voi nähdä skriptin toiminnon, kun sitä suoritetaan tcl-komentotulkissa. Kuviossa skriptiä suoritetaan verkkolaitteen *flash*-muistista, skriptiä voi myös suorittaa USB-muistitikulta, joka on kiinnitetty verkkolaitteen USB-porttiin. USB-muistilta suorittaessa tulee huomioida USB-muistin formaatti johon USB-muisti on alustettu. Ciscon verkkolaitteet tukevat FAT-muistiformaattia. USB-muistin voi formatoida oikeaan formaattiin laittamalle se kiinni reitittimeen ja ajamalla komento EXEC-tilassa *”format usbflash1”*, tällöin verkkolaite alustaa itse USB-muistin oikeaan formaattiin. Formatoinnin seurauksena USB-muistista häviää kaikki tiedostot.

```
router#tclsh flash:fturt.tcl
Please enter the hostname of this router: R1
Please enter the domain name: con7.local
Please enter logging buffered size 4096-2147483647(bytes): 4096
Please enter the aaa authentication list name: R1
Please enter the username for this router: ville.korhonen
Please enter the password for ville.korhonen: korhonen.ville
Please enter the enable password: vkor
Please enter the WAN IP address: 192.168.200.200
Please enter the WAN netmask: 255.255.255.0
Please enter the default route next-hop IP address: 192.168.200.1
Please enter the LAN IP address: 192.168.20.1
Please enter the LAN netmask: 255.255.255.0
Please enter the LAN interface: FastEthernet0/1
Please enter the LAN interface description: LAN
Configuring the DHCP service...
Please enter the DNS Server IP address: 192.168.200.250
Please enter the excluded IP address from DHCP assignments - separated by spaces: 192.168.20.1
Please enter the DHCP Pool Name: LAN
Please enter the DHCP lease time(days): 7
Activating SSH ...
Please enter the NTP server address: 192.168.200.250
Please enter your timezone name: FIN
Please enter your timezone(Hours offset from UTC): +2
Router configuration done!

R1#
```

Kuvio 3. Reitittimen asetukset skriptillä

Yllä olevassa kuviossa Ciscon reitittimeen ajetaan skriptin mukana ns. perusasetukset. Eli asetukset jolla luodaan tarvittavat toiminnot, jotta reititintä voidaan hallinnoida etänä, sekä luodaan lähiverkko johon voidaan liittää verkon aktiivilaitteita ja työasemia. Skriptissä suoritetaan myös toimintoja ”taustalla”, eli reitittimeen laitetaan asetuksia ilman että käyttäjän tarvitsee erikseen hyväksyä niitä. Liitteessä 1, on ajettavan skriptin sisältö. Sisältö itsessään on yksinkertainen, eikä siinä suoriteta vertailua laitteessa oleviin asetuksiin. Skriptissä on otettu huomioon laitteen tietoturva – asetuksia, kuten poistettu käytöstä tarpeettomia palveluita (http-hallintayhteys). Vertailemalla liitteen 2 ja 3 tietoja, voimme nähdä skriptin jälkeiset muutokset reitittimellä. Skriptin suorittamisen jälkeen laitteen asetuksiin on muuttunut, laitteen nimi, lisätty käyttäjätunnus, DHCP-palvelu lähiverkolle, lisätty NTP-palvelin, ja tärkeimpänä lisätty WAN-rajapinta, johon osoitettu reitti jolla mahdollistetaan liikennöinti lähiverkosta ulos Internettiin.

Skriptissä komennolla ”*puts -nonewline ”…”* ”, tcl-komentotulkki syöttää asetetun tekstin komentotulkille. Komennolla ”*flush stdout*”, tiputtaa kaikki tiedot jotka on puskuroitu komentotulkille. Komennolla ”*set … [gets stdin]*”, asetetaan halutulle muuttujalle arvoksi puskurilta saatu arvo, tässä tapauksessa arvo on se joka on syötetty komentoriville. Komennolla ”*ios_config ”…”* ”, muutetaan järjestelmän asetuksia. Järjestelmän asetuksia muuttaessa käytettävät komennot ovat samoja komentoja kun *EXEC*- ja *config*- tilassa käytetyt komennot.

5.3 Allekirjoitettu TCL skripti

5.3.1 Yleistä

Allekirjoitetulla TCL skriptillä voidaan varmistaa skriptin eheys ja aitous. Skriptiä ei ole siirtovaiheessa kukaan ulkopuolinen päässyt muokkaamaan. Allekirjoitettu skripti ei itsessään ole salattu. Allekirjoitettu Tcl skripti pitää sisällään alkuperäisen skriptin lähdekoodin, sekä allekirjoitetun version skriptistä. Jotta allekirjoitetun skriptin voi suorittaa Ciscon reitittimellä, täytyy siinä olla *crypto*-lisenssi. Tämän voi varmistaa syöttämällä reitittimen komentoriville *exec*-tilassa *sh version*, reititin tulostaa komentoriville laitteen IOS-version, jonka perässä on merkintä */k9*. K9-merkintä tarkoittaa

että reitittimessä on aktiivinen *crypto*-lisenssi. Lisenssin avulla reitittimelle voi luoda *IPSEC*-tunneleita, ja reitittimelle voi syöttää x509-sertifikaatin mukaisia salausavaimia. *Crypto*-lisenssin avulla Tcl-komentotulkki pystyy käsittelemään digitaalisesti allekirjoitettua tietoa.

Palvelin jolla allekirjoitettu skripti luodaan, tulee olla asennettuna *openssl*, jolla voidaan luoda digitaalisesti allekirjoitettu Tcl skripti, sekä muut tarvittavat salaiset- ja julkiset avaimet. Tcl-skriptien käsittelyyn *expect*-paketti, jonka avulla voidaan luoda ja muokata Tcl-skriptejä. Jotta reitittimen Tcl-komentotulkki pystyy käsittelemään allekirjoitettua tieto, täytyy se muuttaa binäärimuodosta hex-muotoon. Tätä varten palvelimelle tulee asentaa *xxd*-asennuspaketti. Jokaisesta asennuspaketista tulisi tietoturvallisesta näkökulmasta olla viimeisimmät saatavilla olevat versiot asennettuna, tällä voidaan varmistaa että mahdollisia haavoittuvuuksia ei pääse syntymään digitaalisesti allekirjoitettuun tiedostoon. Jokaisen asennuspaketin version saa selville, laittamalla komentoriville ko. toiminnon komennon, ja sen perään-*version*, poikkeuksena openssl komennossa ei tarvitse laitta viivaa *version* komennon eteen (ks. Kuvio 4).

```
[ville.korhonen@server openssl]$ openssl version
OpenSSL 1.0.1e-fips 11 Feb 2013
[ville.korhonen@server openssl]$ expect -version
expect version 5.45
[ville.korhonen@server openssl]$ xxd -version
xxd V1.10 27oct98 by Juergen Weigert
[ville.korhonen@server openssl]$
```

Kuvio 4. Asennuspakettien versiot

5.3.2 Varmenteiden luominen palvelimella

Palvelimena toteutuksessa käytettiin Centos 7-palvelinta, joka on kaupallisen Red Hat Enterprise linuxiin pohjautuva GNU/Linux-käyttöjärjestelmän jakelupaketti. Palvelimelle luotiin käyttäjätunnus *ville.korhonen*, joka lisättiin admin-ryhmään *wheel*. Näin käyttäjällä on oikeus suorittaa järjestelmään muokkaavia komentoja komennolla *sudo*, sekä mahdollista lukea ja lisätä tiedostoja järjestelmän kansioihin. Palvelinta ei

tulisi koskaan hallita *root*-tunnuksella, koska *root*-tunnusta ei voi seurata kuka sitä on käyttänyt. Kun käyttäjä suorittaa komentoja omalla tunnuksellaan, jää siitä merkintä palvelimen lokitietoihin, lisäksi *wheel*-ryhmän käyttäjä voi suorittaa komentoja *root*-tunnuksella vaihtamalla käyttäjän komennolla *su-root*. Näin käyttäjä voi suorittaa toimintoja *root*-oikeuksilla, tästä etuna lokiin jää merkintä kuka on suorittanut komentoja *root*-tunnuksella.

Palvelimelle luodaan *openssl*-hakemisto *etc*-hakemiston alle, kun kansio on lisätty *etc*-hakemistoon ei sitä pääse ulkopuoliset ja normaalit käyttäjät muokkaamaan. Openssl-hakemistoon luodaan yksityinen sekä julkinen avain. Yksityinen avain luodaan komennolla *sudo openssl genrsa-out privkey.pem 2048*. Komennolla saadaan 2048 bitin suuruinen avaintiedosto *pem* muodossa (ks. Kuvio 5).

```
[ville.korhonen@server openssl]$ sudo openssl genrsa -out privkey.pem 2048
Generating RSA private key, 2048 bit long modulus
............+++
....+++
e is 65537 (0x10001)
[ville.korhonen@server openssl]$
```

Kuvio 5. Yksityisen avaimen luominen

Yksityistä avainta ei saa koskaan luovuttaa kenellekään, ja se tulisi säilyttää rajatussa ympäristössä johon ei ole pääsyä verkkoyhteyden kautta. Julkiset avaimet allekirjoitetaan yksityisellä avaimella, jos yksityistä avainta ei säilytetä suojatussa paikassa, mahdollistaa se tietoturvariskin jonka avuilla digitaalisesti allekirjoitetut tiedoston voidaan avata.

Julkinen avain luodaan komennolla *sudo openssl rsa-in privkey.pem-pubout-out pubkey.pem* (ks.Kuvio 6). Julkinen avain luodaan yksityisestä avaimesta, ja sitä voidaan jakaa muille osapuolille.

```
[ville.korhonen@server openssl]$ sudo openssl rsa -in privkey.pem -pubout -out pubkey.pem
writing RSA key
[ville.korhonen@server openssl]$
```

Kuvio 6. Julkisen avaimen luominen

Seuraavaksi luodaan X509-standardin mukainen sertifikaatti. Sertifikaatti luodaan yksityisestä avaimesta. Sertifikaattia luodessa tulee määrittä kuinka pitkään sertifikaatti on voimassa, tiedoston nimi, ja täydentää sen luoneen organisaation tiedot. Sertifikaatti luodaan komennolla *sudo openssl req-new-x509-key privkey.pem-out cert.pem-days 365*. Sertifikaatin nimeksi luotiin *cert.pem* ja se on on 365 päivää voimassa. Sertifikaatin tietoihin tulee myös täydentää maan nimi jossa se on luotu, maakunta, kaupunki, yritys/organisaatio, organisaation yksikön nimi, yleinen nimi joka voi olla sertifikaatin luojan tiedot, sekä sähköpostiosoite (ks. Kuvio 7).

```
[ville.korhonen@server openssl]$ sudo openssl req -new -x509 -key privkey.pem -out cert.pem -
days 365
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [XX]:FI
State or Province Name (full name) []:MF
Locality Name (eg, city) [Default City]:JKL
Organization Name (eg, company) [Default Company Ltd]:FDF
Organizational Unit Name (eg, section) []:IT
Common Name (eg, your name or your server's hostname) []:VKO
Email Address []:ville.korhonen@server.con7
[ville.korhonen@server openssl]$
```

Kuvio 7. X509-sertifikaatin luominen

Seuraavaksi allekirjoitetaan Tcl-skripti DER PKCS#7 formaattiin binäärimuodossa. Tcl-skripti allekirjoitetaan komennolla *sudo openssl smime-in ping_traceroute_v1-out ping_traceroute_v1.pk7-signer cert.pem-inkey privkey.pem-outform DER-binary*. Tcl-skripti *ping_trceroute_v1* allekirjoitetaan *cert.pem* sertifikaatilla ja yksityisellä avaimella *privkey.pem* (ks. Kuvio 8). Tuloksena saadaan *ping_traceroute_v1.pk7*, joka on DER PKCS#7-formaatissa binäärimuodossa.

```
[ville.korhonen@server openssl]$ sudo openssl smime -sign -in ping_traceroute_v1 -out ping_tr
aceroute_v1.pk7 -signer cert.pem -inkey privkey.pem -outform DER -binary
[ville.korhonen@server openssl]$
```

Kuvio 8. Tcl-skriptin allekirjoittaminen

Kun allekirjoitettu skripti on luotu, tulee se vielä tarkastaa luodulla sertifikaatilla. Tarkastuksella varmistutaan siitä, että tiedosto luomisessa ei tapahtunut ylimääräisiä

muutoksia, ja allekirjoitettu tiedosto vastaa alkuperäistä tiedostoa. Varmistuksella saadaan varmuus tiedoston eheydestä. Tarkastaminen suoritetaan komennolla *sudo openssl smime-verify-in ping_traceroute_v1.pk7-Cafile cert.pem-inform DER-content ping_traceroute_v1* (ks.Kuvio 9). Komennolla verrataan allekirjoitetun tiedoston sisältöä alkuperäiseen tiedostoon, jossa varmenteena käytetään luotua sertifikaattia *cert.pem*. Jos tarkastaminen suoritetaan onnistuneesti, saadaan tuloksena tiedoston sisältö tulostettuna komentoriville, ja todennus tarkastuksen onnistumisesta.

```
[ville.korhonen@server openssl]$ sudo openssl smime -verify -in ping_traceroute_v1.pk7 -CAfil
e cert.pem -inform DER -content ping_traceroute_v1
proc validate {args} {
foreach address $args {
        set output [exec "ping $address"]
        puts $output
        if { [regexp {.!!!!} $output]} {
        set output [exec "traceroute $address"]
        puts $output
        }
}
}
validate 192.168.200.100 192.168.200.200 192.168.200.240

Verification successful
[ville.korhonen@server openssl]$
```

Kuvio 9. Allekirjoitetun skriptin eheyden varmistaminen

Seuraavaksi muutetaan allekirjoitettu skripti binäärimuodosta heksadesimaaliseen muotoon. Lisäksi heksadesimaaliseen muotoon lisätään alkuun *#Cisco Tcl signature V1.0*, sekä jokaisen rivin alkuun risuaita(#) merkintä. Näin Tcl-komentotulkki reitittimellä tulkitsee tiedoston allekirjoitettuna skriptinä. Jos reitittimellä ei ole tarvittavia lisenssejä digitaalisesti allekirjoitetun tiedoston purkamiseen, käsittelee Tcl-komentotulkki risuaidalla merkityt rivit kommentti-riveiksi, ja näin ollen ei ota niitä huomioon skriptiä suorittaessa. Tiedosto ei voi muokata sudo oikeuksilla, käyttäjän tulee suorittaa komento *root*-tunnuksella. Käyttäjätunnuksen voi vaihtaa komennolla *su* root, tämän jälkeen syötetään root-tunnuksen salasana, *root*-tunnuksesta voidaan vaihtaa takaisin alkuperäiseen käyttäjään komennolla *exit*. Allekirjoitettu tiedosto muutetaan heksadesimaalimuotoon komennolla *xxd-ps ping_traceroute_v1.pk7 > ping_traceroute_v1.hex* (ks. Kuvio 10).

```
[ville.korhonen@server openssl]$ su root
Password:
[root@server openssl]# xxd -ps ping_traceroute_v1.pk7 > ping_traceroute_v1.hex
[root@server openssl]#
```

Kuvio 10. Tiedoston muuttaminen heksadesimaaliin

Seuraavaksi kun alkuperäinen skripti on allekirjoitettu, muutettu heksadesimaalimuotoon ja lisätty tarvittavat kommentit, yhdistetään alkuperäinen skripti ja heksadesimaaliin muutettu ja kommentoitu tiedosto yhteen. Tiedostot yhdistetään komennolla *cat ping_traceroute_v1 ping_traceroute_v1 > ping_traceroute_v1.tcl*. Komento tulee ajaa *root* -käyttäjätunnuksella (ks. Kuvio 11). Tuloksena saadaan liitteen 1. mukainen tiedosto.

```
[ville.korhonen@server openssl]$ su root
Password:
[root@server openssl]# cat ping_traceroute_v1 ping_traceroute_v1.hex_sig > ping_traceroute_v1
.tcl
[root@server openssl]# ls -l ping_traceroute_v1*
-rw-r--r-- 1 root root  262 Jan  9 14:40 ping_traceroute_v1
-rw-r--r-- 1 root root 3431 Jan  9 14:50 ping_traceroute_v1.hex
-rw-r--r-- 1 root root 3517 Jan  9 14:54 ping_traceroute_v1.hex_sig
-rw-r--r-- 1 root root 1687 Jan  9 14:43 ping_traceroute_v1.pk7
-rw-r--r-- 1 root root 3779 Jan  9 14:58 ping_traceroute_v1.tcl
[root@server openssl]#
```

Kuvio 11. Tcl-skriptin luominen reitittimelle

Yllä olevassa kuviossa näkyy kaikki skriptistä luodut tiedostot. Tcl-päätteellä oleva tiedosto on lopullinen tiedosto joka siirretään reitittimelle. Pk7-tiedosto on allekirjoitettu tiedosto alkuperäisestä skriptistä. Hex-päätteiset tiedostot ovat heksadesimaaliin muutettuja tiedostoja allekirjoitetusta tiedostosta, hex_sig-tiedosto on heksadesimaalitiedosto johon on lisätty tarvittavat kommentit, jotta reitittimen Tcl-komentotulkki suorittaa allekirjoitetun heksadesimaaliin muutetun tiedoston.

Lopuksi Tcl-skripti siirretään käyttäjän *ville.korhonen* kotihakemistoon, josta se voidaan noutaa verkon yli käyttäen suojattua scp-yhteyttä. Tiedosto siirretään komennolla *cp /etc/openssl/ping_traceroute_v1.tcl /home/ville.korhonen* (ks. Kuvio 12).

```
[ville.korhonen@server openssl]$ cp /etc/openssl/ping_traceroute_v1.tcl /home/ville.korhonen
```

Kuvio 12. Tiedoston siirtäminen kotihakemistoon

5.3.3 Allekirjoitetun skriptin suorittaminen reitittimellä

Kun tarvittava skripti on luotu ja allekirjoitettu, lisätään reitittimelle sertifikaatti jolla voidaan avata allekirjoitettu skripti. Otetaan reitittimeen yhteys konsolikaapelilla, tai salattu etäyhteys verkon yli. Molemmissa tapauksissa reitittimelle tulee olla ennalta asetettuna käyttäjätunnus jolla voidaan oikeuttaa käyttäjä kirjautumaan reitittimelle. Jokaiselle käyttäjälle jonka tarvitsee muokata reitittimen asetuksia, tulisi olla yksilölliset tunnukset reitittimelle. Tämä on voitu toteuttaa luomalla jokaiselle käyttäjälle oma paikallinen tunnus reitittimellä, tai vaihtoehtoisesti käyttää ulkopuolista AAA-palvelinta käyttäjien tunnistamiseen reitittimellä. Ulkopuolisen tunnustautumisen etuna on käyttäjätietojen rekisteröinti palvelimella, näin voidaan seurata kuka on kirjautunut reitittimelle, mitä muutoksia henkilö on tehnyt reitittimellä.

Kirjaudutaan reitittimelle käyttäjällä jolla on oikeudet muokata reitittimen *crypto*-asetuksia. Tämän jälkeen suoritetaan *config*-tilassa komento *crypto pki trustpoint server*. Komennolla lisätään CA – palvelin nimellä *server*. Samalla CA – palvelimelle määritetään kuinka sille annetaan sertifikaatti. Komennolla *enrollment* terminal, määritetään sertifikaatin kopioimenen kopioi-liitä periaatteella komentoriviltä. Eli käyttäjä kopioi sertifikaatin tiedostosta tiedot, ja liittää ne komentoriville. Sertifikaatti syötetään komennolla *crypto pki authenticate server*, jossa server on CA-palvelimen nimi (ks. Kuvio 13). Tämän jälkeen reititin pyytää liittämään sertifikaatin, kun sertifikaatti on syötetty lisätään se perään sana *quit* ja painetaan *enter*.

```
RT.BLUE(config)#crypto pki trustpoint server
RT.BLUE(ca-trustpoint)#enrollment terminal
RT.BLUE(ca-trustpoint)#crypto pki authenticate server

Enter the base 64 encoded CA certificate.
End with a blank line or the word "quit" on a line by itself

MIIDyzCCArOgAwIBAgIJAKP6P1+ucrvhMA0GCSqGSIb3DQEBCwUAMHwxCzAJBgNV
BAYTAkZJMQswCQYDVQQIDAJNRjEMMAoGA1UEBwwDSktMMQwwCgYDVQQKDANGREYx
```

Kuvio 13. CA -palvelimen lisääminen reitittimelle

Kun sertifikaatti on syötetty, pyytää reititin varmistamaan sertifikaatin eheyden ja aitouden. Eheys varmistetaan vertaamalla reitittimen laskemaa MD5, tai SHA-1 tarkistussummaa alkuperäiseen sertifikaatista laskettuun tarkastussummaan. Jos tarkastussummat täsmäävät, voidaan sertifikaatti hyväksyä reitittimellä. Näin saadaan varmistettua, että sertifikaattia ei ole kukaan päässyt muokkaamaan siirron yhteydessä, ja että sertifikaatti on eheä (ks. Kuvio 14).

```
Fk2ViMiY8thG6ipPsuSsk9KWFJwGeXgsuOFV+zbK6bvAAtduDEbHlXI6vtGe2TPd
gZ3mTuXe9/s6Ppt85zSb

Certificate has the following attributes:
       Fingerprint MD5: 62726DC0 840FA416 5B7316A4 D75ABC2C
      Fingerprint SHA1: 6A17C939 39293142 3126203F 29E821D4 BDFB2AAA

% Do you accept this certificate? [yes/no]: yes
Trustpoint CA certificate accepted.
% Certificate successfully imported

RT.BLUE(config)#
```

Kuvio 14. Sertifikaatin hyväksyminen

Sertifikaatin syöttämisen jälkeen, määritetään Tcl-komentotulkille allekirjoitettujen Tcl-skriptien suorittamiseen tarvittavat asetukset. Komentotulkille määritetään luotettu CA-palvelin ja varmenne, joita se käyttää digitaalisesti allekirjoitettujen skriptien suorittamiseen. Komennolla *scripting tcl trustpoint name server* määritetään käytettäväksi CA-palvelimeksi *server* (ks. Kuvio 15). Komennolla *scripting tcl secure-mode* määritetään allekirjoitusten todentaminen (ks. Kuvio 15). Lopuksi määritetään kuinka Tcl-komentotulkki käsittelee jos allekirjoitettua tiedostoa ei pystytä varmenta-

maan annetun palvelimen varmenteella. Komennolla *scripting tcl trustpoint untrusted terminate* asetetaan Tcl-komentotulkki hylkäämään suoritettava Tcl-skripti, jos sitä ei pystytä suorittamaan annetulla varmenteella (ks. Kuvio 15).

```
RT.BLUE(config)#scripting tcl trustpoint name server
RT.BLUE(config)#scripting tcl secure-mode
RT.BLUE(config)#scripting tcl trustpoint untrusted terminate
RT.BLUE(config)#
```

Kuvio 15. Tcl-komentotulkin määritykset

Ennen skriptin kopioimista palvelimelta, tarkastetaan vielä että reitittimellä olevan CA-palvelimen tiedot ovat oikeat ja niissä ei ole poikkeavuuksia. Tiedot voidaan varmistaa komennolla *show crypto pki trustpoints* (ks. Kuvio 16), komento suoritetaan *EXEC*-tilassa. Tulosteena saadaan varmenteen luoneen organisaation tiedot, ja varmenteen sarjanumero.

```
RT.BLUE#show crypto pki trustpoints
Trustpoint server:
    Subject Name:
    e=ville.korhonen@server.con7
    cn=VKO
    ou=IT
    o=FDF
    l=JKL
    st=MF
    c=FI
          Serial Number (hex): 00A3FA3F5FAE72BBE1
    Certificate configured.


RT.BLUE#
```

Kuvio 16. CA -palvelinten tarkastaminen

Seuraavaksi kopioidaan allekirjoitettu Tcl-skripti palvelimelta reitittimelle. Tiedostojen siirtämiseen palvelimen ja reitittimen välillä tulisi aina käyttää suojattua yhteyttä, jossa käyttäjä tunnistetaan palvelimella. Näin tiedon siirtäminen säilyy turvallisena, ja etäkäytöstä jää palvelimen lokitietoihin merkintä kuka palvelimelle on kirjautunut ja

mistä. Tiedosto voidaan kopioida palvelimelta komennolla *copy scp:ping_traceroute_v1.tcl flash0:/tcl/* (ks. Kuvio 17). Komennossa määritetään mistä tiedosto kopioidaan, mikä on tiedoston nimi palvelimella, ja mihin tiedosto tallennetaan reitittimellä. Tiedoston siirtämiseen käytetään suojattua SCP-yhteyttä, kopioitava tiedosto on allekirjoitettu Tcl-skripti, ja tiedosto tallennetaan Flash-muistille kansioon *tcl*. Komennon jälkeen reititin pyytää palvelimen IP-osoitteen tai domain-nimen, tunnuksen jolla palvelimelle kirjaudutaan, kohdetiedoston nimen reitittimellä, sekä salasana käyttäjätunnukselle (ks. Kuvio 17).

```
RT.BLUE#copy scp:ping_traceroute_v1.tcl flash0:/tcl/
Address or name of remote host []? 192.168.200.250
Source username [admin]? ville.korhonen
Destination filename [/tcl/ping_traceroute_v1.tcl]?
Password:
 Sending file modes: C0754 3779 ping_traceroute_v1.tcl
!
3779 bytes copied in 6.136 secs (616 bytes/sec)

RT.BLUE#
```

Kuvio 17. Tiedon siirtäminen palvelimelta reitittimelle

Jos annetut tiedot ovat oikein ja palvelin tunnistaa käyttäjän tiedot, hakee reititin palvelimelta pyydetyn tiedoston. Tulosteena saadaan siirtoon käytetty aika, siirrettävän tiedoston koko, ja tiedon siirtämiseen käytetty nopeus bit/sek.

Kun tarvittavat määritykset on tehty reitittimelle, ja suoritettava skripti on siirretty reitittimen muistille, voidaan digitaalisesti allekirjoitettu skripti suorittaa komennolla *tclsh flash0:/tcl/ping_traceroute_v1.tcl* (ks. Kuvio 18).

```
RT.BLUE#tclsh flash0:/tcl/ping_traceroute_v1.tcl

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.168.200.100, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/4 ms

Type escape sequence to abort.
Tracing the route to 192.168.200.100
VRF info: (vrf in name/id, vrf out name/id)
  1 192.168.200.100 0 msec *  0 msec

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.168.200.200, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms

Type escape sequence to abort.
Tracing the route to 192.168.200.200
VRF info: (vrf in name/id, vrf out name/id)
  1 192.168.200.200 4 msec *  0 msec

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.168.200.240, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/1 ms

Type escape sequence to abort.
Tracing the route to 192.168.200.240
VRF info: (vrf in name/id, vrf out name/id)
  1 192.168.200.240 0 msec *  0 msec

RT.BLUE#
```

Kuvio 18. Allekirjoitetun Tcl-skriptin suorittaminen reitittimellä

Jos Tcl-skriptin allekirjoituksen tarkastus suoritetaan onnistuneesti, suoritetaan skripti Tcl-komentotulkilla normaalisti. Reitittimen lokitiedoista voimme nähdä, että skriptin allekirjoitus on vastannut reitittimellä olevaa varmennetta.

```
Jan  9 13:15:44.482: CRYPTO_PKI: (A0021)chain cert was anchored to trustpoint server, and chain validation result was: CRY
PTO_VALID_CERT
Jan  9 13:15:44.482: CRYPTO_PKI: Success on PKCS7 verify!
Jan  9 13:15:44.482: CRYPTO_PKI: Rcvd request to end PKI session A0021.
Jan  9 13:15:44.482: CRYPTO_PKI: PKI session A0021 has ended. Freeing all resources.
Jan  9 13:15:44.482: CRYPTO_PKI: unlocked trustpoint server, refcount is 0
RT.BLUE#
```

Kuvio 19. Lokitieto allekirjoitetun skriptin suorittamisesta

Jos Tcl-skriptin allekirjoitus ei vastaa reitittimellä olevan varmenteen allekirjoitusta, ei Tcl-komentotulkki suorita skriptiä (ks. **Error! Reference source not found.**). Näin voi tapahtua kun reitittimelle on syötetty uusi varmenne edellisen vanhennettua.

```
RT.BLUE#tclsh flash0:/tcl/ping_traceroute_v1.tcl
Invalid Signature
RT.BLUE#
```

Kuvio 20. Skriptin suorittaminen väärällä varmenteella

Jos reitittimen Tcl-komentotulkille on määritelty *safe-execute*, suorittaa Tcl-komentotulkki skriptin suojatussa tilassa, jos se ei vastaa reitittimellä olevaa varmennetta. Kokeiluna reitittimelle syötettiin uusi varmenne, joka oli kirjoitettu uudella yksityisellä avaimella. Tuloksena *safe-execute*-tilassa suoritettua skriptiä ei voitu ajaa, koska skripti vaatii oikeudet tehdä suorituksia *exec*-tilassa, mm. tämä toiminnollisuus on rajoitettu reitittimellä jotta vahinkoa ei pääse syntymään.

```
RT.BLUE#tclsh flash0:/tcl/ping_traceroute_v1.tcl
invalid command name "exec"
    while executing
"exec "ping $address""
    (procedure "validate" line 3)
    invoked from within
"validate 192.168.200.100 192.168.200.200 192.168.200.240"
    (file "flash0:/tcl/ping_traceroute_v1.tcl" line 11)

RT.BLUE#
```

Kuvio 21. Skriptin suorittaminen "safe-execute" tilassa

Suoritetulla skriptillä testataan yhteydellisyys verkossa oleviin laitteisiin, skriptiin voidaan määrittää useita laitteita joihin yhteydellisyys halutaan testata, tässä tapauksessa kolmeen ennalta määrättyyn IP-osoitteeseen.

```
proc validate {args} {
foreach address $args {
        set output [exec "ping $address"]
        puts $output
        if { [regexp {.!!!!} $output]} {
        set output [exec "traceroute $address"]
```

```
        puts $output
        }
}
}
validate 192.168.200.100 192.168.200.200 192.168.200.240
```

Skriptissä luodaan proseduuri *validate* jolle määritetään argumentti *args*, argumentille on ennalta määritelty arvot *192.168.200.100 192.168.200.200 192.168.200.240*. *Foreach*-komennolla luodaan lista *address* joka hakee argumentin *args* arvoja. Komennolla *set output [exec "ping $address"]* määritetään argumentti *output,* jonka sisällä suoritetaan reitittimen *exec*-tilassa *ping* komento. Joka lähettää *args* listassa oleville arvoille ICMP *echo request*-paketteja. Jotka tulostetaan komentoriville komennolla *puts $output*. Jos tulosteessa on *".!!!!"*, suoritetaan ko. arvolle *traceroute*-komento, jolla selvitetään mitä reittiä protokollan paketit kulkevat määrättyyn kohteeseen. Joka lopuksi tulostetaan komentoriville. Kun skriptis suoritetaan Tcl-komentotulkilla, saadaan tuloksena onnistuneista ja epäonnistuneista *ping*-komennoista, sekä tulostus *traceroute* komennosta niihin laitteisiin, jotka vastasivat ICMP *echo request*-paketteihin (ks. Kuvio 18).

5.4 Saatavuuden seuraaminen

Järjestelmän toimivuuden kannalta ja saatavuuden seuraamaiseksi, tiedonsiirtojärjestelmien ylläpitäjät joutuvat jatkuvasti seuraamaan verkossa tapahtuvia muutoksia. Tällaisia muutoksia voivat olla mm. verkko-yhteyksien katkeaminen. Isoissa verkkoympäristöissä seuranta joudutaan toteuttamaan käyttämällä mm. SNMP-protokollaa, kaikissa ympäristöissä tämä ei ole mahdollista järjestelmien rajoituksista ja tietoturvasta johtuen. Pienemmissä ympäristöissä seuranta voidaan toteuttaa automatisoidusti käyttäen järjestelmässä olevia toimintoja palveluiden saatavuuden seuraa-

miseksi. Cisco IOS-laitteissa saatavuuden seuraaminen ja siitä raportointi, voidaan toteuttaa asettamalla SLA-seuranta verkko-osoitteelle. Saatavuuden raportointi voidaan toteuttaa käyttämällä EEM-toimintoa, jolla voidaan lähettää sähköpostia halutuille vastaanottajille.

Ensiksi asetetaan reitittimelle ”*ip sla*”-palvelun seuranta, jolla seurataan verkko-osoitetta *ICMP-echo* viestityksellä.

router(config)#ip sla 10

router(config-ip-sla)#icmp-echo 192.168.200.20

router(config-ip-sla)#timeout 5000

router(config-ip-sla)#frequency 5

Verkkolaitteelle luodaan ”*ip sla”* -profiili, joille annetaan numeroksi 10. Määritetään profiilille miten haluttua palvelua seurataan, komennolla ”*icmp-echo 192.168.200.20”* määritetään, että palvelua seurataan *”ping”* -viestityksillä, lisäksi määritetään seurattavan kohteen IPv4 -osoite. Seurannalle määritetään toistuvuus, sekä aikakatko. Aikakatko määritetään komennolla ”*timeout”* -johon aika syötetään millisekunteina. Toistuvuus määritetään komennolla ” *frequency*”, jolle arvo syötetään sekunteina. Lopuksi määritetään *”ip sla”* -profiilille ajatus, jolloin profiili aktivoidaan. Ajastuksen voi määrittää alkamaan haluttuna vuorokauden aikana, tai vaihtoehtoisesti profiilin voi aktivoida toimimaan jatkuvasti.

router(config)#ip sla schedule 10 life forever start-time now

Yllä olevalla komennolla luotu profiili asetetaan toimimaan jatkuvasti, ja alkamaan heti kun komento on syötetty.

Seuraavaksi luodaan ”ip sla”-profiilille seuranta, jolla voidaan raportoida tapahtumien muutoksia *syslog*-palvelimelle. Toiminnolla myös aktivoidaan EEM-palvelu.

router(config)#track 1 ip sla 10 reachability
router(config-track)# delay down 10 up 15

Komennolla "*track 1 ip sla 10 reachability*" määritetään *track*-profiili tunnuksella 1, jolla seurataan *ip sla*-profiilia 10, ja sen saatavuutta. *Track*-profiiliin määritetään myös se raja-arvot, kuinka kauan saatavuus voi olla alhaalla tai ylhäällä, ennen kuin siitä lähetetään *syslog*-viesti. Ylläolevassa komennossa jos palvelu ei vastaa kymmeneen sekuntiin, niin siitä lähetetään viesti. Jos palvelu on alhaalla olon jälkeen saatavilla yhtäjaksoisesti viisitoista sekuntia, niin lähetetään siitä viesti.

Seuraavaksi määritetään EEM-palvelimelle sähköpostipalvelimen IPv4-osoite, sähköpostiosoite johon viestit lähetetään, ja lähettäjän sähköpostiosoite.

router(config)#event manager environment _email_server 192.168.200.250
router(config)#event manager environment _email_to ville.korhonen@con7.local
router(config)#event manager environment _email_from RT.BLUE@con7.local

Sähköpostipalvelimen osoitteeksi määriteltiin "*192.168.200.250"*, joka on toteutusympäristössä oleva linux-sähköpostipalvelin. Vastaanottajaksi määrittelin oman sähköpostiosoitteen, ja lähettäjäksi määriteltiin reitittimen nimi. Kun EEM-palvelimelle on määritelty tarvittavat sähköposti osoitteet ja yhteystiedot, lisätään EEM-sovellus (applet). Sovellus määrätään aktivoitumaan *track*-seurannan lähettäessä *syslog*-viestin saatavuuden katkeamisesta.

router(config)#event manager applet email_server_unreachable

router(config-applet)#event track 1 state down

router(config-applet)#action 1.0 syslog msg "Server 200.20 unreachable!"

router(config-applet)#action 1.1 cli command "enable"

router(config-applet)#action 1.2 cli command "del /force flash0:/server_unreachable"

router(config-applet)#action 1.3 cli command "show clock | append flash0:/server_unreachable"

router(config-applet)#action 1.4 cli command "show ip arp 192.168.200.20 | append flash0:/server_unreachable"

router(config-applet)#action 1.5 cli command "show ip route 192.168.200.20 | append flash0:/server_unreachable"

router(config-applet)#action 1.6 cli command "show interface GigabitEthernet0/0 | append flash0:/server_unreachable"

router(config-applet)#action 1.7 cli command "more flash0:/server_unreachable"

router(config-applet)#action 1.8 mail server "$_email_server" to "$_email_to" from "$_email_from" subject "Server Unreachable: ICMP-Echos Failed" body "$_cli_result"

router(config-applet)#action 1.9 syslog msg "Server unreachable: email has been send!"

Kun EEM-sovellus saa seuraamansa herätteen *syslog*-palvelulta, aktivoi se sovelluksessa määritetty toiminnot järjestyksessä. Aluksi sovellus määriteltiin lähettämään *syslog*-viesti, joka ilmoitti että seurattava verkko-osoite ei ole enää saatavilla. Seuraavaksi sovellus asetetaan *"privileged EXEC"*-tilaan, jossa voidaan suorittaa haluttuja komentoja. Sovellus seuraavaksi poistaa komennolla *"del /force flash0:/server_unreachable"*-laitteen *flash*-muistista tekstitiedoston *"server_unreachable"*, jos se on tallennettuna. *Force*-toiminnolla komennossa tiedosto poistettaan vaikka sitä jokin muu sovellus käyttäisi. Komennoilla *action 1.3-1.6* tallennetaan laitteen *flash*-muistille tiedostoon *"server_unreachable"*, tarvittavia tietoja jotka auttavat järjestelmän ylläpitäjää vianselvityksessä. Komennolla *"show clock | append flash0:/server_unreachable"* tulostetaan tiedostoon kellonaika, jolloin sovellus suoritettiin. Komennolla *"show ip arp 192.168.200.20 | append flash0:/server_unreachable"* tulostetaan tiedostoon IP-osoitetta vastaava MAC – osoite. Komennolla *"show ip route 192.168.200.20 | append flash0:/server_unreachable"* tulostetaan tiedostoon viimeinen tiedetty reitti kohde – osoitteeseen, ja tallennetaan se tiedostoon. Komennolla *"show interface GigabitEthernet0/0 | append flash0:/server_unreachable"* tulostetaan laitteen WAN – rajapinnan tiedot, joista selviää mm. onko ko. rajapinnan häiriöitä. Komennolla *action 1.7 cli command "more flash0:/server_unreachable"* tulostetaan tekstitiedostossa olevat tiedot komentoriville. Lopuksi komennolla *"action 1.8 mail server "$_email_server" to "$_email_to" from "$_email_from" subject "Server Unreachable: ICMP-Echos Failed" body "$_cli_result""* lähetetään ennalta määriteltyyn sähköpostipalvelimeen viesti, jossa vastaanottaja ja lähettäjän on määritelty aiemmin arvoille *"_email_to"* ja *"_email_from"*. Sähköpostiviestin sisältö on saman kuin *"server_unreachable"*-tekstitiedoston sisältö. Komennolla *"action 1.9 syslog msg*

"Server unreachable: email has been send!"” lähetetään *syslog*-viesti, jolla kerrotaan että tapahtuneesta on lähetetty onnistuneesti sähköposti.

```
RT.BLUE#sh event manager policy registered
No.  Class     Type    Event Type          Trap  Time Registered          Name
1    applet    user    track               Off   Sun Feb 7 19:27:40 2016   email_server_unreachable
 track 1 state down
 maxrun 20.000
 action 1.0 syslog msg "Server 200.5 unreachable!"
 action 1.1 cli command "enable"
 action 1.2 cli command "del /force flash0:server_unreachable"
 action 1.3 cli command "show clock | append flash0:server_unreachable"
 action 1.4 cli command "show ip arp 198.168.200.5 | append flash0:server_unreachable"
 action 1.5 cli command "show ip route 198.168.200.5 | append flash0:server_unreachable"
 action 1.6 cli command "show interface GigabitEthernet0/0 | append flash0:server_unreachable"
 action 1.7 cli command "more flash0:server_unreachable"
 action 1.8 mail server "$_email_server" to "$_email_to" from "$_email_from" subject "Server Unreachable: ICMP-Echos Failed"
body "$_cli_result"
 action 1.9 syslog msg "Server unreachable: email has been send!"

RT.BLUE#
```

Kuvio 22. Rekisteröity EEM-sovellus

EEM-sovelluksen rekisteröinti varmistetaan komennolla *”show event manager policy registered”*. Tuloksena saadaan kaikki rekisteröidyt EEM-sovellukset laitteella (ks. Kuvio 22).

```
RT.BLUE#sh track 1
Track 1
  IP SLA 10 reachability
  Reachability is Up
    1 change, last change 00:00:38
  Delay up 15 secs, down 10 secs
  Latest operation return code: OK
  Latest RTT (millisecs) 1
  Tracked by:
    EEM applet email_server_unreachable
RT.BLUE#sh ip sla summary
IPSLAs Latest Operation Summary
Codes: * active, ^ inactive, ~ pending

ID           Type        Destination       Stats       Return      Last
                                           (ms)        Code        Run
---------------------------------------------------------------------
*10          icmp-echo   192.168.200.20    RTT=1       OK          4 seconds ago
```

Kuvio 23. Yhteys kunnossa (track, ip sla)

Ylläolevassa kuvioissa nähdään asetettujen “*ip sla”* ja “*track 1”* profiilien välinen toiminta. *”IP sla”*-profiilin *”icmp-echo”*-kysely suoritettiin edellisen kerran neljä sekuntia

sitten onnistuneesti. *”Track”*-profiilista näemme että yhteys palvelimeen on kunnossa, ja saatavilla kun tilan on *”UP”*. ”Track”-profiilista näemme myös, että profiilia seuraa luotu EEM-sovellus. Alla olevassa kuviossa tilanne jossa yhteys palvelimeen on katkennut. *”Track”*-profiilin tila on *”Down”,* ja *”ip sla”*-profiilin tila on *”Timeout”.*

```
RT.BLUE#sh track 1
Track 1
  IP SLA 10 reachability
  Reachability is Down
    2 changes, last change 00:00:00
  Delay up 15 secs, down 10 secs
  Latest operation return code: Timeout
  Tracked by:
    EEM applet email_server_unreachable
RT.BLUE#sh ip sla summary
IPSLAs Latest Operation Summary
Codes: * active, ^ inactive, ~ pending

ID           Type        Destination       Stats       Return      Last
                                           (ms)        Code        Run
---------------------------------------------------------------------
*10          icmp-echo   192.168.200.20    -           Timeout     14 seconds ag
                                                                   o
```

Kuvio 24. Yhteys poikki (track, ip sla)

Laitteen *syslog*-viesteistä voimme nähdä onko sovellus aktivoitunut, kun palvelin ei ole enää ollut saatavilla.

```
Feb  7 18:36:19.791: %TRACK-6-STATE: 1 ip sla 10 reachability Up -> Down
Feb  7 18:36:19.791: %HA_EM-6-LOG: email_server_unreachable: Server 200.20 unreachable!
Feb  7 18:36:38.543: %HA_EM-6-LOG: email_server_unreachable: Server unreachable: email has been send!
RT.BLUE#
```

Kuvio 25. EEM-lokiviestit

Ylläolevista lokiviesteistä voimme nähdä että EEM-sovellus on aktivoitunut kuten piti, ja että asetetut *sysylog*-viestit on lähetetty. Seuraavaksi tarkastetaan komennolla *”dir” enable*-tilassa, onko EEM-sovellus luonut laitteen *flash*-muistiin tekstitiedoston (ks. Kuvio 26).

```
RT.BLUE#dir
Directory of flash0:/

   1  -rw-    97114876  Aug 13 2014 02:52:52 +02:00  c2900-universalk9-mz.SPA.154-1.T1.bin
   2  -rw-        3064  Aug 13 2014 03:05:30 +02:00  cpconfig-29xx.cfg
   3  drw-           0  Nov 11 2015 12:18:20 +02:00  tcl
   8  drw-           0  Aug 13 2014 03:06:00 +02:00  ccpexp
 249  -rw-        2464  Aug 13 2014 03:07:44 +02:00  home.shtml
   4  -rw-        1796   Feb 7 2016 20:36:20 +02:00  server_unreachable
```

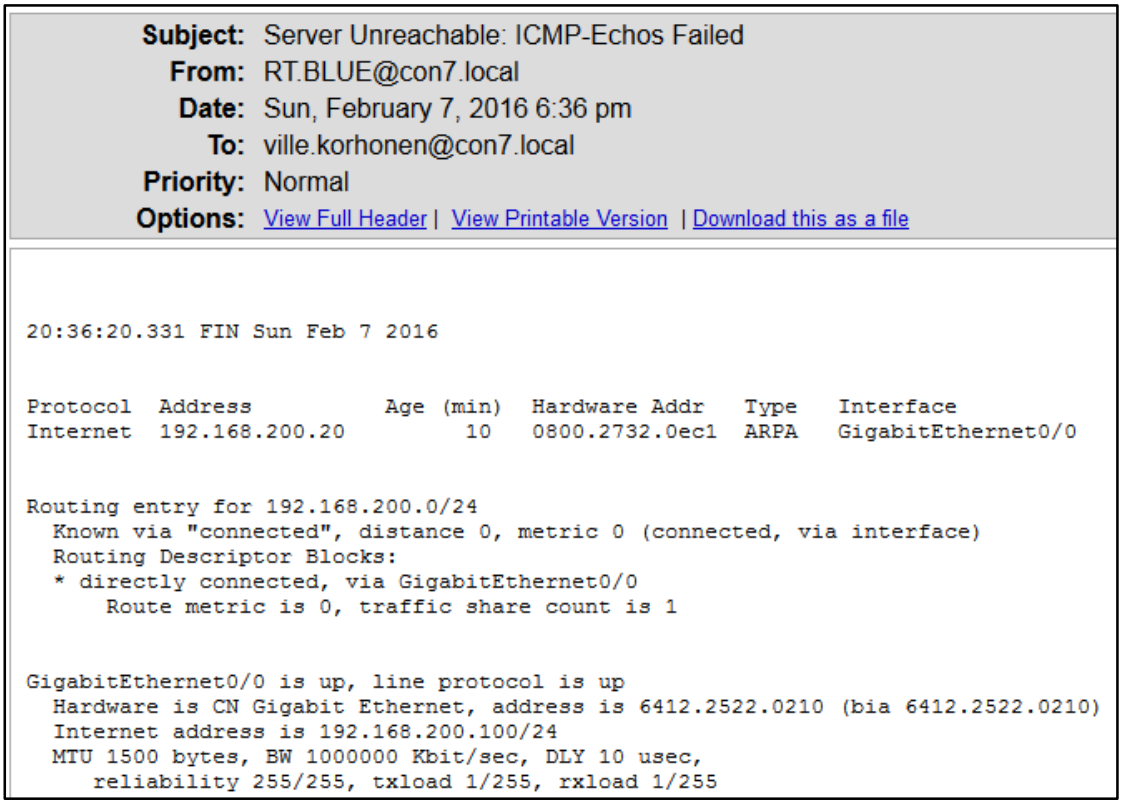
Kuvio 26. Laitteen flash-muistissa olevat tiedostot

Komennolla ”more flasho:/server_unreachable” voimme tulostaa tekstitiedoston sisällön komentoriville (ks. Kuvio 27).

```
RT.BLUE#more flash0:/server_unreachable
20:36:20.331 FIN Sun Feb 7 2016


Protocol  Address          Age (min)  Hardware Addr   Type   Interface
Internet  192.168.200.20         10   0800.2732.0ec1  ARPA   GigabitEthernet0/0


Routing entry for 192.168.200.0/24
  Known via "connected", distance 0, metric 0 (connected, via interface)
  Routing Descriptor Blocks:
  * directly connected, via GigabitEthernet0/0
      Route metric is 0, traffic share count is 1


GigabitEthernet0/0 is up, line protocol is up
  Hardware is CN Gigabit Ethernet, address is 6412.2522.0210 (bia 6412.2522.0210)
  Internet address is 192.168.200.100/24
  MTU 1500 bytes, BW 1000000 Kbit/sec, DLY 10 usec,
     reliability 255/255, txload 1/255, rxload 1/255
```

Kuvio 27. Tekstitiedoston sisältö

Lopuksi tarkastetaan että tekstitiedoston sisältö on myös lähetetty laitteelta sähköpostilla vastaanottajalle.



Kuvio 28. Sähköpostiviestin sisältö

Ylläolevasta kuvioista voimme todeta, että sähköpostiviestin sisältö on sama kuin laitteen *flash*-muistilla olevalle tekstitiedostolla. Näin verkon ylläpitäjä saa nopeasti tiedon kun tiettyyn verkossa olevaan laitteeseen menetetään yhteys.

6 Tulokset

Opinnäytetyön tuloksena saatiin esimerkkejä siitä, kuinka pienen lähiverkon automatisointia ja siitä raportointia voidaan toteuttaa, ottaen huomioon verkon tietoturvallisuudelle asetetut vaatimukset. Esimerkkien avulla lähiverkossa voidaan toteuttaa laitteen käyttöönotto yhteneväisesti skriptin avulla, vaikka laitteen käyttöönoton tekisi eri henkilö. Tällöin laitteiden asetukset tulevat yhtenevästi, ja verkkolaiteiden asetuksissa tulee mahdollisimman vähän ristiriitaisuuksia. Etuna kun verkkolaite otetaan käyttöön skriptin avulla, saadaan varmistettua laitteen tietoturvallisuus, kun laite kovennetaan käyttöönoton yhteydessä. Haasteena laitteen käyttöönottamisessa skriptin avulla kohdattiin eri järjestelmäversioiden välillä. Verkkoympäristö koostuu usein eri järjestelmäversioita, jotka tukevat eri ominaisuuksia tcl-ohjelmointikielessä. Parhaimman tuloksen sai aikaiseksi, kun skriptin teki versiopäivitykseltään vanhimman laitteen mukaan, tällöin sama skripti toimi kaikissa järjestelmäversioissa. Etuna on, että sama skripti toimi kaikille joka työllistää huomattavasti vähemmän. Haittana, ei päästä hyödyntämään uusimpien järjestelmäversioiden ominaisuuksia, jossa on paljon hyödyllisiä ominaisuuksia.

Toteutuksessa saatiin hyvä esimerkki kuinka skriptin voi suorittaa verkkolaitteella tietoturvallisesti. Allekirjoitetussa skriptissä voidaan varmistaa, että tietoa ei ole kukaan muuttanut siirron yhteydessä. Haittana tämä tuo skriptien luomiseen muutamana vaiheen lisää. Skriptit täytyy luoda, jonka jälkeen ne lähetetään allekirjoitettavaksi CA-palvelimelle, jonka jälkeen ne siirretään verkkolaitteelle. CA-palvelimella vaiheet voidaan automatisoida skripteillä hyvinkin helposti. Skriptin siirrossa tulee huomioida KATAKRIn vaatimus tietoturvalliselle tiedonsiirrolle. Lähtökohtaisesti oletetaan että skriptin luoja, ei tee skriptiä CA-palvelimella. Jolloin skripti tulisi siirtää verkossa vahvasti salattua kanavaa pitkin, tai kuljettaa salatulla siirtomateriaalilla CA-palvelimelle. Tämä luo haasteita verkon ylläpitäjille, kun skriptien luomiselle ja niiden toimittamiselle tulee lisää vaiheita. Jotka KATAKRIn vaatimuksesta, riippuen materiaalin luokituksesta, tulee erikseen kirjata suojatun materiaalin siirto luokitellun tiedonsiirtoympäristön ulkopuolella.

Toteutuksessa on esimerkkinä, kuinka voidaan Cisco IOS verkkolaitteelle luoda EEM-sovelluksia, jotka aktivoituvat halutuista herätteistä. Tässä toteutuksessa ohjelman aktivoituessa lähetettiin käyttäjälle sähköpostia. Tämä on kustannustehokas ja nopea tapa saada tieto palvelun saatavuuden muutoksesta. Kaikissa yrityksissä ei ole erillistä henkilöä seuraamaan lokitietoja jatkuvasti, ja henkilöt jotka eivät aktiivisesti kerkeä seuraamaan lokitietoa, on sähköpostin lähettäminen tehokas tapa saada ylläpitohenkilö kiinni. Haittana on, että sähköpostia ei pystytä lähettämään salattuna tässä toteutuksessa, johtuen verkkolaitteiden järjestelmäversioista.

Skriptaamista voidaan toteutuksen perusteella pitää hyvin kustannustehokkaana tapana saada verkkolaitteista irti haluamaansa tietoa, sekä nopeana tapana muokata verkkolaiteiden asetuksia. Jos järjestelmänylläpitäjä ei ole aiemmin ohjelmoinut, tai ohjelmointi on ollut hyvin vähäistä, voi haasteeksi muodostua uuden ohjelmointikielen opetteleminen.

7 Pohdinta ja jatkokehitys

Työ aloitettiin helmikuussa 2015 toimeksiannon yhteydessä. Toimeksiantoon tutustuin tieteellisten artikkelien, uutisointien ja alan kirjallisuuteen perehtyen. Pyrin teoriaosuudessa ottamaan huomioon nykypäivän uutisoinnit tietoverkkojen toimivuudesta, sekä vertaamaan niitä luettuun teoriaan. Ja tämän pohjalta luomaan näkymään siitä, kuinka nykypäivän tietoverkot tulisi toteuttaa automatisoidusti. Luetusta teoreettisesta materiaalista oli suuresti hyötyä käytännön toteutuksen kanssa. Kirjoittamisen aloitin Huhtikuussa 2015, ja se jatkui aina käytännön toteutukseen saakka Marraskuuhun 2015. Teoriaosuuden kirjoittaminen oli suhteellisen helppoa, teoriaosuus kuitenkin muuttui kun työtä rajasin käytännön toteutuksen myötä. Tämä hankaloitti hieman teorian kirjoittamista.

Materiaali tietoverkon automatisointiin löytyi niukasti, pääasiallinen materiaali tietoverkkojen automatisoinnista löytyi Internetistä julkaistuista kirjoista, sekä tieteellisistä artikkeleista. Tietoverkon automatisointiin ohjelmallisesti tukeviin ohjelmointikieliin löytyi runsaasti materiaalia Internetin julkaisuina, kuin myös kirjallisena materiaalina. Suurin osa automatisointiin liittyvästä materiaalista oli englannin kielellä.

Tämä aiheutti kirjoittamisessa hankaluuksia aiheeseen liittyvän terminologian kanssa. Kun kaikille vastaaville sanoille ei löydy suomenkielessä vastinetta.

Toteutusympäristön sain valmiiksi Marraskuussa 2015, jolloin pääsin testaamaan laitteiden toimintojen suorittamista ohjelmallisesti. Toteutuksessa suurimmaksi päänvaivaksi muodostui tehtävien skriptien tarkkuus, yksi ylimääräinen välilyönti sai aikaiseksi sen, että koko skripti ei toiminut.

Opintojen aikaisemmilta kursseilta saatu osaaminen ja teoreettinen tietopohja olivat tärkeitä teorian kirjoittamisessa, sekä käytännön toteutuksessa. Koulutusohjelma on tuonut hyvät näkökulmat siihen kuinka tietoverkon hallinnointi tulisi tehdä, ottaen huomioon tietoturvallisuuden tiedonsiirrossa, ja sen käsittelyssä.

Opinnäytetyössä saatiin hyviä esimerkkejä kuinka tietoverkon laitteita voidaan automatisoida, uskon työstäni olevan hyötyä jatkossa toimeksiantajalle.

Kehittäminen

Verkon automatisointia voisi laajentaa huomattavasti hyödyntäen Cisco IOS – laitteiden ominaisuuksia. Laitteilta voisi kerätä enemmän tietoa yhdelle keskitettyyn palvelimeen, jossa kerätyn tiedon pohjalta tehdään automaattisia päätöksiä. Palvelin voisi luoda itse tarvittavan skriptin ennalta asetettuihin määrityksiin perustuen. Herätteitä voisi kerätä samalta verkkolaitteelta useasta lähteestä, ja verrata näitä tietoja keskenään, sekä tehdä niiden pohjalta automaattisia toimenpiteitä. Samalla herätteistä voisi tehdä kootun raportin, joka lähetetään tietyin väliajoin verkon ylläpitäjälle tarkasteltavaksi. Skripteihin voisi lisätä debuggaus-toimintoja, jos skriptiä ei päästä suorittamaan onnistuneesti, niin laite loisi tästä raportin.

Lähteet

Blair R. 2010. TCL Scripting for Cisco IOS: A Guide to Building and Modifying Tcl Scripts to Automate Network Administration Tasks. Viitattu 11.10.2015. http://www.jamk.fi/fi/Palvelut/kirjasto/Etusivu/, Nelli-portaali, Books24x7.

Göransson, P. & Black, C. 2014. Software Defined Networks: A Comprehensive Approach. Viitattu 17.1.2016.http://www.jamk.fi/fi/Palvelut/kirjasto/Etusivu/, Nelli-portaali, Books24x7.

Hartig, O. 2014. Miksi liikenne ei kääntynyt varareitille? Elisa ymmällään poikkeuksellisesta tilanteesta. Viitattu 18.6.2015. http://www.tivi.fi/Uutiset/2014-11-27/Miksi-liikenne-ei-k%C3%A4%C3%A4ntynyt-varareitille-Elisa-ymm%C3%A4ll%C3%A4%C3%A4n-poikkeuksellisesta-tilanteesta-3150968.html

Jones, D.2005. Automating Network Management and Compliance. Viitattu 8.6.2015. http://www.realtimepublishers.com/book.php?id=29

Juntunen, E. 2014. Tämä sormen paksuinen kaapeli kaatoi satojentuhansien suomalaisten nettiyhteydet. Viitattu 18.6.2015. http://www.hs.fi/kotimaa/a1417064608487

KATAKRI 2015. 2015a. Katakri 2015 Tietoturvallisuuden auditointityökalu viranomaisille. Viitattu 1.9.2015. http://www.defmin.fi/files/3165/Katakri_2015_Tietoturvallisuuden_auditointityokalu_viranomaisille.pdf

KATAKRI 2015. 2015b. Katakri 2015 – Tietoturvallisuuden auditointityökalu viranomaisille. Viitattu 10.9.2015. http://www.defmin.fi/puolustushallinto/puolustushallinnon_turvallisuustoiminta/katakri_2015_-_tietoturvallisuuden_auditointityokalu_viranomaisille

Lappalainen J. 2010. Perusteita skriptauksesta. Viitattu 1.9.2015. https://webapps.jyu.fi/wiki/display/opentvt/Skriptaus

ONF. 2012. Software-Defiened Networking: The New Form for Networks. Viitattu 31.1.2016. https://www.opennetworking.org/images/stories/downloads/sdn-resources/white-papers/wp-sdn-newnorm.pdf

Puolustusvoimien tehtävät. Pääesikunnan viestintäosasto 2013.Viitattu 15.6.2015. http://www.puolustusvoimat.fi/fi/Puolustusvoimat/Etusivu/?urile=wcm%3Apath%3A/su%20puolustusvoimat.fi/Puolustusvoimat.fi/Puolustusvoimat/Etusivu/, Perustieto, Puolustusvoimien tehtävät.

Tcl Developper. 2015a.About Tcl/Tk Language. Viitattu 24.10.2015. https://www.tcl.tk/about/language.html

Tcl Developper. 2015b. Foreach. Viitattu 23.12.2015. http://wiki.tcl.tk/1018

Tcl Developper. 2015c. Tcl 8.5.18 Tcl cmd if. Viitattu 22.12.2015. http://www.tcl.tk/man/tcl8.5/TclCmd/if.htm

2015a. Viestintäviraston verkkosivut. Ohje hyväksyttävien yhdyskäytäväratkaisujen suunnitteluperiaatteista ja ratkaisumalleista. Viitattu 17.1.2016. https://www.viestintavirasto.fi/attachments/Yhdyskaytavaratkaisuohje.pdf

2015b. Viestintäviraston verkkosivut. Viestintäviraston NCSA-toiminnon hyväksymät salausratkaisut. Viitattu 17.1.2016. https://www.viestintavirasto.fi/attachments/tietoturva/NCSA_salausratkaisut.pdf

Liitteet

Liite 1. Reitittimen asetukset -skripti

```
puts -nonewline "Please enter the hostname of this router: "
flush stdout
set hostname [ gets stdin ]
puts -nonewline "Please enter the domain name: "
flush stdout
set domain [ gets stdin ]
ios_config "ip domain name $domain"
ios_config "hostname $hostname"
puts -nonewline "Please enter logging buffered size 4096-2147483647(bytes): "
flush stdout
set logbuf [ gets stdin ]
ios_config "logging buffered $logbuf"
ios_config "ip cef"
ios_config "no ip domain lookup"
ios_config "no ip http server"
ios_config "no logging console"
ios_config "no logging monitor"
ios_config "no cdp run"
ios_config "service password-encryption"
ios_config "no service config"
ios_config "line con 0" "logging synchronous"
ios_config "aaa new-model"
puts -nonewline "Please enter the aaa authentication list name: "
flush stdout
set aaalistname [ gets stdin ]
ios_config "aaa authentication login $aaalistname local"
puts -nonewline "Please enter the username for this router: "
flush stdout
```

```
set username [ gets stdin ]
puts -nonewline "Please enter the password for $username: "
flush stdout
set password [ gets stdin ]
ios_config "username $username privilege 15 password $password"
puts -nonewline "Please enter the enable password: "
flush stdout
set enable [ gets stdin ]
ios_config "enable secret $enable"
puts -nonewline "Please enter the WAN IP address: "
flush stdout
set wan_ip [ gets stdin ]
puts -nonewline "Please enter the WAN netmask: "
flush stdout
set wan_netmask [ gets stdin ]
ios_config "interface FastEthernet0/0" "ip address $wan_ip $wan_netmask"
ios_config "interface FastEthernet0/0" "no shutdown"
puts -nonewline "Please enter the default route next-hop IP address: "
flush stdout
set nxthop [ gets stdin ]
ios_config "ip route 0.0.0.0 0.0.0.0 $nxthop"
puts -nonewline "Please enter the LAN IP address: "
flush stdout
set lan_ip [ gets stdin ]
puts -nonewline "Please enter the LAN netmask: "
flush stdout
set lan_msk [ gets stdin ]
puts -nonewline "Please enter the LAN interface: "
flush stdout
set lan_int [ gets stdin ]
ios_config "interface $lan_int" "ip address $lan_ip $lan_msk"
ios_config "interface $lan_int" "no shutdown"
```

```
puts -nonewline "Please enter the LAN interface description: "
flush stdout
set lan_des [ gets stdin ]
ios_config "interface $lan_int" "description $lan_des"
puts "Configuring the DHCP service..."
puts -nonewline "Please enter the DNS Server IP address: "
flush stdout
set dns_ip [ gets stdin ]
puts -nonewline "Please enter the excluded IP address from DHCP assignments - separated by spaces: "
flush stdout
set excluded [ gets stdin ]
foreach i $excluded {
    ios_config "ip dhcp excluded-address $i"
}
puts -nonewline "Please enter the DHCP Pool Name: "
flush stdout
set dhcp_name [ gets stdin ]
puts -nonewline "Please enter the DHCP lease time(days): "
flush stdout
set dhcp_lt [ gets stdin ]
ios_config "ip dhcp pool $dhcp_name" "network $lan_ip $lan_msk"
ios_config "ip dhcp pool $dhcp_name" "default-router $lan_ip"
ios_config "ip dhcp pool $dhcp_name" "dns-server $dns_ip"
ios_config "ip dhcp pool $dhcp_name" "lease $dhcp_lt"
puts "Activating SSH ..."
ios_config "line vty 0 4" "transport input ssh"
ios_config "line vty 0 4" "transport output ssh"
ios_config "crypto key generate rsa general-keys modulus 2048"
ios_config "ip ssh version 2"
puts -nonewline "Please enter the NTP server address: "
flush stdout
```

```
set ntpaddr [ gets stdin ]
ios_config "ntp server $ntpaddr"
puts -nonewline "Please enter your timezone name: "
flush stdout
set tznam [ gets stdin ]
puts -nonewline "Please enter your timezone(Hours offset from UTC): "
flush stdout
set tzone [ gets stdin ]
ios_config "clock timezone $tznam $tzone"
puts "Router configuration done!"
```

Liite 2. Reitittimen asetuksen ennen skriptiä

```
router#sh run
Building configuration...

Current configuration : 984 bytes
!
version 12.4
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption
!
hostname router
!
boot-start-marker
boot-end-marker
!
!
no aaa new-model
memory-size iomem 5
no ip icmp rate-limit unreachable
ip cef
```

!

!

!

!

no ip domain lookup

!

multilink bundle-name authenticated

!

!

!

!

!

!

!

!

!

!

!

!

!

!

!

!

!

!

!

!

!

!

!

!

archive

log config

hidekeys

!

```
!
!
!
ip tcp synwait-time 5
!
!
!
!
interface FastEthernet0/0
 no ip address
 shutdown
 duplex auto
 speed auto
!
interface FastEthernet0/1
 no ip address
 shutdown
 duplex auto
 speed auto
!
interface FastEthernet1/0
 no ip address
 shutdown
 duplex auto
 speed auto
!
ip forward-protocol nd
!
!
no ip http server
no ip http secure-server
!
```

no cdp log mismatch duplex

!

!

!

!

!

!

control-plane

!

!

!

!

!

!

!

!

!

!

line con 0

exec-timeout 0 0

privilege level 15

logging synchronous

line aux 0

exec-timeout 0 0

privilege level 15

logging synchronous

line vty 0 4

login

!

!

end

router#

Liite 3. Reitittimen asetuksen skriptin jälkeen

R1#sh run

Building configuration...

Current configuration : 1647 bytes

!

version 12.4

service timestamps debug datetime msec

service timestamps log datetime msec

service password-encryption

!

hostname R1

!

boot-start-marker

boot-end-marker

!

logging buffered 4096

no logging console

no logging monitor

enable secret 5 1s21B$5nAL6corvE60FEmC9IpWT/

!

aaa new-model

!

!

aaa authentication login R1 local

!

!

aaa session-id common

memory-size iomem 5

```
clock timezone FIN 2
no ip icmp rate-limit unreachable
ip cef
!
!
no ip dhcp use vrf connected
ip dhcp excluded-address 192.168.20.1
!
ip dhcp pool LAN
  network 192.168.20.0 255.255.255.0
  default-router 192.168.20.1
  dns-server 192.168.200.250
  lease 7
!
!
no ip domain lookup
ip domain name con7.local
!
multilink bundle-name authenticated
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
```

```
!
!
!
!
!
!
!
!
username ville.korhonen privilege 15 password 7
141C1D19040B242E2A66253C2E1F02
archive
 log config
  hidekeys
!
!
!
!
ip tcp synwait-time 5
ip ssh version 2
!
!
!
!
interface FastEthernet0/0
 ip address 192.168.200.200 255.255.255.0
 duplex auto
 speed auto
!
interface FastEthernet0/1
 description LAN
 ip address 192.168.20.1 255.255.255.0
 duplex auto
```

speed auto

!

interface FastEthernet1/0

no ip address

shutdown

duplex auto

speed auto

!

ip forward-protocol nd

ip route 0.0.0.0 0.0.0.0 192.168.200.1

!

!

no ip http server

no ip http secure-server

!

no cdp log mismatch duplex

no cdp run

!

!

!

!

!

!

control-plane

!

!

!

!

!

!

!

!

```
!
!
line con 0
 exec-timeout 0 0
 privilege level 15
 logging synchronous
line aux 0
 exec-timeout 0 0
 privilege level 15
 logging synchronous
line vty 0 4
 transport input ssh
 transport output ssh
!
ntp server 192.168.200.250
!
end

R1#
```

Liite 4. Allekirjoitettu TCL skripti

```
proc validate {args} {
foreach address $args {
                set output [exec "ping $address"]
                puts $output
                if { [regexp {.!!!!} $output]} {
                set output [exec "traceroute $address"]
                puts $output
                }
```

}

}

validate 192.168.200.100 192.168.200.200 192.168.200.240

#Cisco Tcl Signature V1.0

#3082069306092a864886f70d010702a0820684308206800201013l0b3009

#06052b0e03021a0500300b06092a864886f70d010701a08203cf308203cb

#308202b3a003020102020900a3fa3f5fae72bbe1300d06092a864886f70d

#01010b0500307c310b3009060355040613024649310b300906035504080c

#024d46310c300a06035504070c034a4b4c310c300a060355040a0c034644

#46310b3009060355040b0c024954310c300a06035504030c03564b4f3129

#302706092a864886f70d010901161a76696c6c652e6b6f72686f6e656e40

#7365727665722e636f6e37301e170d3136303130393132333532305a170d

#3137303130383132333532305a307c310b3009060355040613024649310b

#300906035504080c024d46310c300a06035504070c034a4b4c310c300a06

#0355040a0c03464446310b3009060355040b0c024954310c300a06035504

#030c03564b4f3129302706092a864886f70d010901161a76696c6c652e6b

#6f72686f6e656e407365727665722e636f6e3730820122300d06092a8648

#86f70d0101010500382010f003082010a0282010100c102ec0a7800078d

#2e8808487f18df7d6befab5dfb9367af7eab976b38fadc9236cd472d8ded

#57e14f14193c3a5869af5aeb9202570885e67fb294771b05d5414f3d85e3

#aefbdd39d5e1fd28bcdf939cdddc23014aeb1311d2bf05e61336e3c13881

#681e1853fbd5b36fcc280700cecb92919bf0d63817c5165d5b129c5bfdae

#adc0a4448b843027529d5c341b4d9ab4463d97de4e95738114fba0653a89

#7f691bb5f47c044a10b1c8dd657f56fe16880f602323915ba54ac545e386

#cd058aee478283e79ff8cff58b53492d0b0ba466bb5c65d7b1c04ae6e25e

#921add5e08e934c4c6b746f68b51cfd6cdf78a13793e376ad707a560fcf4

#ad4ec4feed476e7b0203010001a350304e301d0603551d0e04160414dcfe

#d424494cbe96bba6bd85a64bfc4c52be2661301f0603551d230418301680

#14dcfed424494cbe96bba6bd85a64bfc4c52be2661300c0603551d130405

#30030101ff300d06092a864886f70d01010b0500038201010079ba706232

#fe7140a7ac4c2bbf0d8e3d5565a84bfb9da8c9ec405f7a2a8b0925aee93e

#2a3a1d43068e2665d1b8c41a3859d10669bdb22de055b4a3d8ddf17fe132

#e45b42c13845d51d911e5712fe27dd80c5029c964fae7463aaf3e8a7ff11

#5bceb405eb3ac29b9ee38c65a759c208fac036c3f159fa4907da0304dd25

#56b514e5291d857501b37f3386ad987c6fd82397cbdefd47c14078cccedd

#f64b799047357400862dee3878c16b6405846bdda8eb6cac5c812da1f41c

#65a62bde850e50ab164d9588c898f2d846ea2a4fb2e4ac93d296149c0679

#782cb8e155fb36cae9bbc002d76e0c46c795723abed19ed933dd819de64e

#e5def7fb3a3e9b7ce7349b3182028c3082028802010130818930 7c310b30

#090603550406130246493 10b300906035504080c024d46310c300a060355

#04070c034a4b4c310c300a060355040a0c03464446310b3009060355040b

#0c024954310c300a06035504030c03564b4f3129302706092a864886f70d

#010901161a76696c6c652e6b6f72686f6e656e407365727665722e636f6e

#37020900a3fa3f5fae72bbe1300906052b0e03021a0500a081d830180609

#2a864886f70d010903310b06092a864886f70d010701301c06092a864886

#f70d010905310f170d313630313039313234333139 5a302306092a864886

#f70d01090431160414baef0e482080cd0236ceaac182668f132c0f4f8730

#7906092a864886f70d01090f316c306a300b06096086480165030401 2a30

#0b06096086480165030401163 00b06096086480165030401023 00a06082a

#864886f70d0307300e06082a864886f70d030202020080300d06082a8648

#86f70d0302020140300706052b0e030207300d06082a864886f70d030202

#0128300d06092a864886f70d0101010500048201007901c8081524f7646d

#2b40c6e742590e7adc7598b8757649eb036c72e8b71da35fd52dd421c31e

#f84edb2102fadffc54d90055062e0a6053cd083fa0cecddd68b03fd8fce4

#8a4d473ada2d831c20240088c4b7b037416d098222e7d24232019198a897

#e09c0cab6929da69350ed6d242cabd1c2395de7760708f1767d07df4b3f6

#3683f9519997a5119eeed4e46e0b961c80dc967e908925dfb8d7cce1566e

#283b979b3ba71a3cda9cc18b6d8c23f190697b6c10bd740066450ca747aa

#b827b81cff3e68ff5223ccc75ecc7c813a01a0532fffd100ce58c4a15cd0

#a545db5e4195fed0338396ebac9f81ef3aa1b2b26ae36926feffc0131a41

#47c558ea3f1bb4

#